On the Theory of the Polynomial Degrees of the Recursive Sets

AUTHOR(S):

SHINODA, JUICHI; SLAMAN, THEODORE A.

CITATION:

SHINODA, JUICHI ...[et al]. On the Theory of the Polynomial Degrees of the Recursive Sets. 数理解析研究所講究録 1988, 644: 90-145

ISSUE DATE:

1988-02

URL:

http://hdl.handle.net/2433/100237

数理解析研究所講究録
第644巻 1988年 90-145

On the Theory of the Polynomial Degrees of the Recursive Sets

JUICHI SHINODA
THEODORE A. SLAMAN

ABSTRACT. There is an interpretation of first order arithmetic in the theory of the PTIME degrees of the recursive sets. There is an interpretation of second order arithmetic in the first order theory of the PTIME degrees. These results characterize the Turing degrees of the first order theories of these structures.

§1. INTRODUCTION

A set of integers R is recursive, if there is an effective method to determine its elements. Given an integer n, the method can be applied, in finitely many steps, to give the answer yes, if n is an element of R, and no, otherwise. A is recursive in B, if there is a method to compute A given a method to compute B. If each of A and B is recursive in the other, then A and B have the same Turing degree.

The Turing degree of A is a measure of the effective information in A. It gives some idea of A's mathematical accessibility. In particular, if A is not recursive, there is a substantial impediment to uncovering information about A.

Similarly, it is possible to give a finer notion for the degree of A in order to discuss finer distinctions in relative computability. In this context, it is more appropriate to study sets of finite strings x in a finite language. Use $|x|$ to denote the length of x.

In Cook [2], A is said to be PTIME computable in B, if there is a polynomial u and a method using B to determine whether a string x belongs to A, such that, for every x, the answer is always found in less than $u(|x|)$ many steps. Here, we will not specify any particular model for computational step, except to assume that the computational method is deterministic. We write $A \leq_p B$ and call the induced degrees the PTIME degrees. Let $\langle REC, \leq_p \rangle$ and $\langle \mathcal{R}, \leq_p \rangle$ be the structures of the PTIME degrees of the recursive sets and of all sets, respectively, with the partial ordering induced from PTIME relative computability.

This paper was prepared while Slaman visited Nagoya University with a fellowship from the Japan Society for the Promotion of Science. In addition, Slaman was supported by Presidential Young Investigator Award DMS-8451748 and N.S.F. research grant DMS-8601856.

In this paper, we will apply recursion theoretic technology to develop some of the metatheory of these structures. In particular, we will calculate the Turing degrees of their first order theories. We show that each of these structures is undecidable.

Let N be the usual structure of arithmetic on $\{0, 1, \dots\}$. The first order language of arithmetic has symbols for addition, multiplication, 0 , 1, variables to refer to integers and quantifiers to range over the integers. The second order language has, in addition, variables referring to sets of integers, a membership symbol indicating that an integer belongs to a set and quantifiers ranging over sets. We will show that there are interpretations of the first order theory of N in the first order theory of $\langle REC, \leq_p \rangle$ and of the second order theory of N in the first order theory of $\langle \mathcal{R}, \leq_p \rangle$.

THEOREM. (1) *There is a recursive function mapping $\varphi \mapsto \varphi^*$ such that, for all first order sentences φ,*

$$N \models \varphi \iff \langle REC, \leq_p \rangle \models \varphi^*.$$

(2) *There is a recursive function $\varphi \mapsto \varphi^*$ such that, for all second order sentences φ in the language of arithmetic*

$$N \models \varphi \iff \langle \mathcal{R}, \leq_p \rangle \models \varphi^*.$$

Using Gödel's Incompleteness Theorem, we can conclude that these structures are undecidable as a corollary.

Since $\langle REC, \leq_p \rangle$ is defined in first order arithmetic and $\langle \mathcal{R}, \leq_p \rangle$ is defined in second order arithmetic, there are canonical interpretations of the theories of the PTIME degree structures in their associated levels of arithmetic. Thus, the theories of $\langle REC, \leq_p \rangle$ and N have the same Turing degree; similarly, the theories of $\langle \mathcal{R}, \leq_p \rangle$ and the structure associated with second order arithmetic have the same Turing degree.

For the moment, we focus our attention on the interpretation of arithmetic in $\langle REC, \leq_p \rangle$. First, we give an interpretation of arithmetic in the theory of a fixed recursive partially ordered set P_N. Second, we give a coding scheme whereby a sequence of degrees $\vec{p}$ defines a subset of REC with a partial ordering $\preceq$. The coding mechanism is analogous to ones that have been successful in substructures of the Turing degrees. It combines ingredients from Harrington-Shelah [4], Harrington-Slaman [5] and Slaman-Woodin [10].

Next, we give a criterion that is definable in $\langle REC, \leq_p \rangle$, to pick out a set sequences $\vec{p}$ that code partial orders canonically containing a copy of P_N. In this step, we adapt a device from Shore [9]. We require that $\vec{p}$ code a partial order that canonically contains a model M of a finite fragment of arithmetic. Further, the partial order coded by $\vec{p}$ contains enough extra structure that the ideal (S) generated by standard part of M is definable in terms of an a exact pair: $(H_0) \wedge (H_1) = (S)$. Further, we arrange that the integers of M are an independent set so that the standard part of M is definable from the parameters H_0 and H_1. But then, "$\vec{p}$ codes a standard model of arithmetic" is a definable property of $\vec{p}$.

Finally, we prove that the set of sequences coding a standard model is not empty by explicitly constructing one.

Our proofs are organized using the priority method from recursion theory. For example, see Soare [11]. In section §2, we give an introduction to the method. In short, we build sets by a recursion in which the action taken during each stage is effective. During each stage, we specify the values of finitely many sets on finitely many strings. This is called specifying a condition. We also impose some constraints on the conditions that will be allowed during future stages. This is called specifying an environment. For each requirement of the theorem being proven, we define a strategy to impose a stage by stage sequence of environments that ensures that the requirement is satisfied in the end. In section §2, we give a sufficient notion of compatibility between families of strategies for there to be a recursive construction that respects a strategy from each family.

In section §3, we use the priority method as formulated in section §2 to classify when an ideal can be represented as the meet of two principal ideals in each of $\langle REC, \leq_p \rangle$ and $\langle \mathcal{R}, \leq_p \rangle$: in the first case, if it has a recursive presentation; in the second case, if it is countable. Our analysis of the existence of exact pairs duplicates work that was done independently and substantially earlier by Ambos-Spies [1].

In section §4, we define P_N, formulate the coding machinery and prove that there is a definable set consisting of codes for standard models of arithmetic.

Sections §5, §6 and §7 are devoted to showing that the set of codes for standard models is not empty. In section §5, we give a sufficient list of requirements. In section §6, we give a family of strategies for each requirement. In terms of the classification of priority methods given in Groszek-Slaman [3], these are all Π_2-strategies, making this an

"infinite injury" construction. However, exploiting special properties of our strategies, we can avoid much of the complexity of the Π_2-priority method. In section §6, we prove, for each requirement, any construction following one of its associated strategies produces sets that satisfy the requirement. In §7, we prove that the strategies are compatible in the sense of section §2. Hence, there is a construction building sets that satisfy all of the requirements.

By organizing our proof in this way, we have attempted to make our presentation as modular as possible. In reading the proof, it is illuminating to look at the simplest strategies from each family and check compatibility. We also recommend that while reading section §2, the reader look for examples in section §3.

In section §8, we draw a further conclusion from the ability to represent models of arithmetic. Following an argument of Nerode-Shore [8], we show that any automorphism of $\langle \mathcal{R}, \leq_p \rangle$ must preserve arithmetic degree on all sufficiently large elements. We end with several questions.

§2. Recursion Theoretic Groundwork

In the following, the reader may choose any particular deterministic model for computation. We will only need to fix some measure of time in order to interpret the notion of a polynomially time bounded procedure. We will use PTIME as a prefix indicating that there is a polynomial u such that the method of evaluation at a given string x uses at most u of the length of x many steps.

We will work exclusively with the object language $\{0,1\}^*$, the set of finite binary strings. For an integer ℓ, let $\{0,1\}^{\leq \ell}$ be the set of strings of length less than or equal to ℓ. We use lower case Roman letters $x, y, \ldots$ to indicate strings. Let $|x|$ denote the length of x. A *predicate* is a subset of $\{0,1\}^*$. We use upper case Roman letters $A, B, \ldots$ to indicate predicates. We will not make any distinction between the predicate X and the function from $\{0,1\}^*$ into $\{0,1\}$ that is equal to 1 exactly on the elements of X. Using a PTIME pairing function, let $\langle x, y \rangle$ denote the string coding the ordered pair of strings x and y. We will assume that $|\langle x, y \rangle|$ is greater than either $|x|$ or $|y|$. Through pairing, we can regard a predicate as an infinite sequence of predicates. Let $X^{(n)}$ denote the nth element in the sequence coded by X; we will refer to $X^{(n)}$ as the nth column of X. We let $X \oplus Y$ denote the PTIME join of X and Y.

We use upper case Greek letters Ψ to indicate Turing functionals. For the most part, Ψ will indicate a PTIME binary valued functional. In this case, $\Psi(Y)$ is the predicate computed by Ψ relative to Y. For a string x, $\Psi(Y,x)$ is this predicate's value at x. We use lower case Greek letters to indicate the computation associated with the evaluation of a functional. For example, $\psi(Y,x)$ is the computation of $\Psi(Y,x)$. By $Y\restriction \psi(Y,x)$, we mean the information used about Y during the computation of $\Psi(Y,x)$. We use u_Ψ to denote the polynomial time bound on the computation of Ψ. It is safe to assume that an index for u_Ψ is uniformly obtained from an index for Ψ. Note that every string in $Y\restriction \psi(Y,x)$ is shorter than $u_\Psi(|x|)$.

We use the priority method from recursive function theory to organize our constructions. The reader may consult Soare [**11**] for a general introduction to proofs of this sort. Here, we give a treatment that is tailored for our applications.

Uniformity. In the most general sense, say that A is uniformly given from B if there is a known way to specify A in terms of B. In the context below, we will speak a recursive function $f(\vec{x})$ being uniformly determined from some parameters $\vec{p}_0$. This means that there is an unspecified recursive function $F(\vec{x},\vec{p})$ such that $f(\vec{x})$ is equal to $F(\vec{x},\vec{p}_0)$. The uniformity of f can be usually be deduced from constructive nature of the proof that it exists.

Conditions and Environments. We build predicates by recursion. The steps of a recursion are called *stages*. During the $s+1$st stage, we will work with finitely many predicates. We say that a predicate is *named*, once we begin to work with it. For each named predicate X, we specify finitely many strings that belong to X and finitely that belong to its complement. During subsequent stages, we extend the definition of X to more strings. Ultimately, we define X on every string. It is helpful to look at this situation in the abstract. We will leave the recursive coding in the next definition to the reader.

2.1. DEFINITION. A *condition* p consists of

(1) A finite collection of names.

(2) A positive integer $\ell_p(X)$, for each X named by p.

(3) A function $p(X)$ into $\{0,1\}$ with domain contained in the set of strings of length less than or equal to $\ell_p(X)$, for each X named in p.

We will refer to $p(X)$ as a *condition on* X. If all of the elements of $\overrightarrow{X}$ are named in p, let $p(\overrightarrow{X})$ be the condition on $\overrightarrow{X}$ induced by p.

2.2. DEFINITION. Say that q *extends* p if for every set X named in p, X is also named in q, the domain of $p(X)$ is contained in that of $q(X)$ and $q(X)$ agrees with $p(X)$ on their common domain.

In addition to specifying finitely many atomic facts about X, we may also impose global constraints on the conditions and their extensions that we allow in the construction of X. A collection of such constraints is called an *environment.* Depending on the context, we will also refer to the set of conditions that fulfill the constraints as an environment. To give a simple example, one environment is the set of conditions p such that $p(X)(x) = 0$ if x is in the domain of $p(X)$ and x is not a string of 0's. If X is assembled from conditions in this environment, then X is contained in $\{0\}^*$.

Forcing. In building a sequence of conditions ordered under extension, if we impose the constraint that the $s+1$st condition must come from the environment $E(s+1)$, we have implicitly ensured that the sets constructed will have some specific properties. When the sequence of environments ensures that the sets constructed will satisfy the formula φ, we say that φ is *forced.* We isolate the forcing relation as applied to individual bounded time computations in a specific environment.

2.3. DEFINITION. Work in the environment E with condition q. Let $\overrightarrow{X}$ denote a finite collection of the sets named by elements of E.

(1) Write $q \geq_E r$ to indicate that r is an extension of q in E.

(2) Say that q *strongly forces* $\Phi(\overrightarrow{X}, x) = i$ if every element of $\overrightarrow{X}$ is named in q, every string queried in $\varphi(\overrightarrow{X}, x)$ is in the domain of $q(\overrightarrow{X})$ and the computation yields answer i. Write $q \Vdash^* \Phi(\overrightarrow{X}, x) = i$. We will similarly speak of q strongly forcing an inequality between functionals or the value of a composition of functionals.

(3) Say that q *forces* $\Phi(\overrightarrow{X}, x) = i$ in E if

$$(\forall r \leq_E q)\left[r \Vdash^* \Phi(\overrightarrow{X}, x) = j \implies j = i\right].$$

Here, q forces $\Phi(\overrightarrow{X}, x) = i$ when i is the only value for $\Phi(\overrightarrow{X})$ at x that is consistent with extending q and staying in E. Write $q \Vdash \Phi(\overrightarrow{X}, x) = i$, leaving E to be understood from the context.

(4) Say that q *decides* $\Phi(\vec{X},x)$, if there is an i such that $q \Vdash \Phi(\vec{X},x) = i$. We say that q decides $\Phi(\vec{X})$ on a set of strings if q decides $\Phi(\vec{X})$ on each element of the set. Similarly, q *strongly decides* $\Phi(\vec{X},x)$ if q strongly forces a value.

If $q \Vdash^* \Phi(\vec{X},x) = i$, then $q \Vdash \Phi(\vec{X},x) = i$ in every E. However, the converse need not apply.

2.4. NOTATION. Let $*$ denote the several possible concatenation operations used to extend conditions. If $p(X)$ is a condition on X, let $p(X)*0$ be the predicate that is evaluated 0 on every string not in the domain of $p(X)$. Given an extension X' of $p(X)$, let $p*X'$ be equal to p except at the Xth coordinate where $p*X'$ is equal to $p(X)*X'$, the natural extension of $p(X)$ by X'.

Requirements and Strategies. A *requirement* is a property to be satisfied by the sets under construction. A strategy is a dynamic method to ensure that the requirement is satisfied. We represent a strategy as a finite collection of *states* and *transition conditions and rules* for changing states. During a construction, we begin with S in its initial state. During stage $s+1$, a strategy is given input parameters: $\mathcal{C}\restriction s$, the state of the construction at the end of stage s and $in(s+1)$, the number of steps it takes to determine $\mathcal{C}\restriction s$ together with the action that has already taken place during the current stage.

S changes state during stage $s+1$, if its transition condition is satisfied with these inputs during that stage. The transition conditions will have the form, if there is a condition satisfying certain properties relative to $\mathcal{C}\restriction s$ that can be found by a search of length monotonically depending on $in(s+1)$, then change state according to the appropriate rule. Thus, if a transition condition is met with input $\mathcal{C}\restriction s$ and $in(s+1)$ then it will also be met for any larger second input.

The strategy returns its new state and $E(s+1)$, the environment it imposes during stage $s+1$. In effect, during each stage, the strategy imposes the constraint that the condition chosen during that stage must be an element of $E(s+1)$. We design the strategy to force the statement of its requirement.

Constructions.

2.5. DEFINITION. (1) A *construction* consists of a recursive sequence of conditions $p(s+1)$ ordered under extension, finite sequences of active strategies $\vec{S}(s+1)$, with for each i less than the length of $\vec{S}$, associated values of $in_i(s+1)$ and state $\sigma_i(s+1)$

for the ith element of $\overrightarrow{S}(s+1)$, and a uniformly recursive sequence $f_0, f_1, \ldots$ of auxiliary functions. The auxiliary functions are optional. For all s and each set X named in $p(s)$, $\{0,1\}^{\leq s}$ is contained in the domain of $p(s)(X)$.

(2) The *state of the construction $\mathcal{C}$ at the end of stage s* is denoted by $\mathcal{C}\restriction s$. It consists of the first s values of p, $\overrightarrow{S}$, $\vec{\sigma}$ together with, for i less than the length of $\overrightarrow{S}$, the first s values for in_i, and the first s values of f_i (where appropriate).

(3) A construction *eventually respects* a strategy S if there is an i and an s_0 such that S is equal to the ith active strategy for every stage greater than or equal to s_0; S is in its initial state during stage s_0; for all $s+1$ greater than or equal to s_0, the stage $s+1$ state of S changes according to S's rule, if S's transition condition is met during stage $s+1$; and $p(s+1)$ is an element of the environment imposed by S during stage $s+1$ given inputs $\mathcal{C}\restriction s$ and $in_i(s+1)$. We say that $\mathcal{C}$ respects S after stage s_0.

(4) A construction is *eventually coherent* relative to strategy S if there is an s_0 such that $\mathcal{C}$ respects S after s_0 and for all $s+1$ greater than or equal to s_0, the following hold. First, the graph of S's input function in is PTIME. Secondly, for all $s+1$ greater than or equal than s_0, either S changes state during stage $s+1$ or $p(s+1)$ is uniformly recursive by a computation of length $in(s+1)$.

(5) A construction is *coherent* if it is eventually coherent for all of its eventually active strategies. Further, the PTIME methods of computing the graph of in_i and $p(s+1)$ in $in_i(s+1)$ many steps are uniformly presented in terms of i and the associated activation stage s_0.

A device analogous to coherence appeared independently in the work of Ambos-Spies [1].

2.6. PROPOSITION. *If $\mathcal{C}$ is a coherent recursive construction, then any name appearing in a condition in $\mathcal{C}$ is associated with a recursive predicate.*

PROOF: Since $\mathcal{C}$ is recursive, the only point to check is that each named set defined on all of $\{0,1\}^*$. This is ensured by clause (1) in 2.5. $\Diamond$

Priority. Using the priority method to organize a construction, strategies are assigned a decreasing priority ranking in order of their activation. When a strategy changes state, all strategies of lower priority are cancelled. Each strategy changes state only finitely

often, so each requirement will eventually be assigned a strategy that is never cancelled. A strategy S must have three properties: for each possible terminal state, S must ensure that any sets constructed within its sequence of environments satisfy its requirement; the environment imposed by S must be compatible with those created by higher priority strategies; the set of strategies associated with the remaining requirements must be dense in the limit environment created by S.

During the $s+1$st stage of a construction, we work with $\mathcal{C} \restriction s$, the state of the construction at the end of stage s; $\vec{S}$, a sequence of strategies in states $\vec{\sigma}$; and an integer OUT. We define $\vec{in}(s+1)$ by letting $in_n(s+1)$ equal OUT, letting $in_i(s+1)$ equal the sum of $in_{i+1}(s+1)$ and the number of stages needed to execute S_{i+1} given inputs $\mathcal{C} \restriction s$ and $in_{i+1}(s+1)$.

2.7. DEFINITION. (1) We say that $\mathcal{C}$, $\vec{S}$, $\vec{\sigma}$, and OUT *define a stage* $s+1$ *environment* if for every e less than the length of $\vec{S}$, S_e does not change state from σ_e given inputs $\mathcal{C}$ and $in_e(s+1)$ as above.

(2) The *defined environment* is the conjunction of the constraints returned by the elements of $\vec{S}$.

(3) The sequence $in_n(s+1), \ldots, in_1(s+1)$ is called the *input sequence.*

2.8. DEFINITION. A finite sequence $\vec{S}$ of n strategies in states $\vec{\sigma}$ is *compatible* if there is a recursive method, given $\mathcal{C}$, $\vec{S}$, $\vec{\sigma}$, and *out* as above, and an integer ℓ, to produce a condition q extending $p(s)$ such that one of the following occurs. Let $\vec{in}$ be the input sequence obtained using the given parameters with OUT equal to the sum of *out* and the number of steps needed to compute q.

(1) There is an element S_i of $\vec{S}$ that changes to a new state τ with these inputs to produce a compatible sequence $\vec{S} \restriction i$ in states $\langle \sigma_1, \ldots, \sigma_{i-1}, \tau \rangle$. Further, the new recursive method of finding conditions referred to above is uniformly determined by the new sequence of states.

(2) q is an element of the defined environment. For all X named in q, $\ell_q(X)$ is greater than or equal to ℓ.

Thus, a sequence of strategies is compatible if there is a recursive method to either change state or extend the current condition up to an arbitrary length while staying in the defined environment. Note that the environment is defined after the condition is

computed.

2.9. DEFINITION. A sequence of families of strategies $\vec{\mathcal{S}}$ is *compatible* if there is a recursive method such that given any length n compatible sequence of strategies $\vec{S}$ with states $\vec{\sigma}$, where each $S_i \in \mathcal{S}_i$, the method produces an element S_{n+1} of $\mathcal{S}_{n+1}$ such that the sequence $\vec{S} * S_{n+1}$ with S_{n+1} in its initial state (1) is uniformly compatible.

2.10. THEOREM. *Suppose that $\vec{\mathcal{S}}$ is a compatible sequence of families of strategies such that each strategy mentioned in $\vec{\mathcal{S}}$ can change state at most finitely often. There is a coherent recursive construction $\mathcal{C}$ that eventually respects some strategy from each $\mathcal{S}_i$.*

PROOF: Define the construction $\mathcal{C}$ by the following recursion. We define an auxiliary function OUT to bridge the transition between stages.

Stage 0. Let $p(0)$ be the empty condition. Set $OUT(0) = 0$. There are no active strategies during this stage.

Stage $s+1$. Let $\mathcal{C} \restriction s$ be the state of the construction at the end of stage s. Let $\vec{S} = \langle S_0, \ldots, S_n \rangle$ be the sequence of active strategies during stage s. Let $\vec{\sigma}(s)$ be the sequence of states these strategies occupied at the end of step s.

We use the given recursive method to extend $\vec{S}$ and $\vec{\sigma}(s)$ by adding S_{n+1} from $\mathcal{S}_{n+1}$ in state (1), respectively.

We execute the following nested recursion. Begin with i equal to $n+1$ and OUT equal to the sum of $OUT(s)$ and the number of steps needed to compute S_{n+1}. Iterate the following procedure until no strategy of index less than or equal to i is seen to change state. In going from one iteration to the next, use the least index for a strategy that changes state to be the next value for i, the sum of the previous value of OUT and the number of steps required to do the previous iteration as the next value for OUT, and $\vec{\sigma} \restriction i-1$ followed by the state S_i changed into as the next sequence of states for $\vec{S} \restriction i$.

(i) Since $\langle S_1, \ldots, S_i \rangle$ is compatible, use the given recursive method to find a condition q extending $p(s)$ (depending on OUT and $\mathcal{C} \restriction s$) such that for every X named in q, $\ell_q(X)$ is greater than $s+1$. Let *out* be the number of steps needed to compute q. *See definition 2.9.*

(ii) Let in_i equal $OUT + out$ and let in_k equal the sum of in_{k-1} and the number of steps needed to execute S_{k+1} given input $\mathcal{C} \restriction s$ and in_k.

Use *final* to refer to values occurring in the iteration in which no strategy changed state. Let n_0 be the final value of i. Define the new sequence of active strategies to equal the initial segment of $\overrightarrow{S}$ of length n_0. All other strategies are cancelled. Define $\vec{\sigma}(s+1)$ to be the sequence consisting of the initial segment of $\vec{\sigma}$ of length $n_0 - 1$ followed by the final state that the n_0th element of $\overrightarrow{S}$ entered. For each k less than or equal to n_0, let $in_k(s+1)$ be the final value of in_k. Let $p(s+1)$ be equal the final value of q.

Set $OUT(s+1)$ to equal the number of steps required to calculate the action taken during stage $s+1$.

First, since $\overrightarrow{S}$ is coherent, the recursive methods referred to above are uniformly presented and total. In each stage, the strategies can change state only finitely often so there is a final iteration. Thus, $\mathcal{C}$ is defined at every stage. Further, the method by which $p(s+1)$ is found ensures that $p(s+1)$ extends $p(s)$ and, for each set X named in $p(s+1)$, $p(s+1)(X)$ is defined on $\{0,1\}^{\leq s+1}$. Hence, $\mathcal{C}$ is a construction in the sense of 2.5.

Fix n. Again, since each strategy can change state only finitely often, there is a last stage in $\mathcal{C}$ when some strategy associated with a strategy of index lower than n changes state. After at most $n-1$ additional stages, some S_n in $\mathcal{S}_n$ is added to the list of active strategies, never to be removed. Let s_0 be the stage when S_n is permanently added to the sequence of active strategies. At the beginning stage s_0, S_n is in its initial state. Let $s+1$ be greater than or equal to s_0. By examination of $\mathcal{C}$, either S_n changes state or $p(s+1)$ is an element of the environment imposed by S_n during stage $s+1$ with inputs $\mathcal{C}\restriction s$ and $in_n(s+1)$. Thus, $\mathcal{C}$ eventually respects S_n.

Now consider the graph of in_n. Let $s+1$ be a stage after s_0. Since no strategy of index less than or equal to n changes state during stage $s+1$, either S_n changes state or $in_n(s+1)$ is the number of steps needed to compute an iteration starting from i up to the point where S_n is tested without first having seen a strategy of higher index change state. In the second case, the value of $in_n(s+1)$ is essentially the number of steps needed to compute it. Thus, the graph of in_n is PTIME. Further, for all $s+1$ after the state of S_n has reached its limit value, the choice of $p(s+1)$ is uniformly decided by a computation of length less than $in_n(s+1)$. Thus, $\mathcal{C}$ is eventually coherent relative to S_n with the correct uniformity property.

This is enough to show that $\mathcal{C}$ is coherent. $\Diamond$

Organization of a Proof. First, we write the statement of the theorem as an infinite family of requirements. Second, we introduce the strategies associated with the requirements. For each strategy, we prove that any coherent construction that respects the strategy produces sets that satisfy its requirement. Next, we show that the families of strategies are compatible. Then, we apply Theorem 2.10 to give a recursive construction executing a strategy for each requirement.

§3. Ideals and Exact Pairs

3.1. Definition. An *upper semi-lattice* is a partially ordered set P with a binary operation $\vee$ mapping X and Y to their least upper bound $X \vee Y$. Call $X \vee Y$ the *join* of X and Y.

If P is an upper semi-lattice, then the join operation can be defined in terms of the partial ordering in P. However, having it as a primitive operation makes the language much more succinct. Our degree structures are upper semi-lattices due to the presence of a PTIME pairing function.

Note that all of the following definitions are first order in $\mathcal{I}$, $\mathcal{J}$ and $\mathcal{K}$.

3.2. Definition. (1) In an upper semi-lattice $\mathcal{D}$, an *ideal* is a set $\mathcal{I}$ that is closed under join and closed downward.

(2) Intersection gives an operation of meet on ideals. Union followed by closure under $\mathcal{D}$'s join gives an operation of join for ideals. Let $\mathcal{I} \wedge \mathcal{J}$ and $\mathcal{I} \vee \mathcal{J}$ denote these, respectively.

(3) Given a K in $\mathcal{D}$, let (K) denote $\{X \mid X \leq_{\mathcal{D}} K\}$. (K) is an ideal. Similarly, if $\mathcal{K}$ is a set of elements in $\mathcal{D}$, let $(\mathcal{K})$ denote the join of the ideals (K) such that K is in $\mathcal{K}$.

(4) If $\mathcal{I}$ is an ideal, then K_0 and K_1 in $\mathcal{D}$ are an *exact pair* over $\mathcal{I}$ if

$$(K_0) \wedge (K_1) = \mathcal{I}.$$

One of the first theorems in the study of the Turing degrees is the Kleene-Post theorem [6] that the Turing degrees do not form a lattice. Ladner [7] used the same construction to show that the PTIME degrees do not form a lattice. In the Turing degrees, Spector [12] gave an abstract version of the Kleene-Post argument to show that there is an exact pair

over every countable ideal. We observe that Spector's argument applies to the PTIME degrees and that the effective version of Spector's argument can be applied in $\langle REC, \leq_p \rangle$. These results were proven independently and substantially earlier by Ambos-Spies [1] We include their proofs to be complete and to give the reader a simple context in which to view the strategies involved. Variants of these strategies will reappear in section §6.

3.3. THEOREM. *For any countable ideal $\mathcal{I}$ in $\langle \mathcal{R}, \leq_p \rangle$, there is an exact pair K_0 and K_1 over $\mathcal{I}$.*

PROOF: Let $F_0, F_1, \ldots$ be a list of the elements of $\mathcal{I}$. Let P denote the collection of conditions naming only K_0 and K_1. Let $\langle p_0, p_1 \rangle$ denote the condition p where $p(K_i)$ is equal to p_i. Order P by extension. For each n, let $\langle q_0, q_1 \rangle \leq_{P_n} \langle p_0, p_1 \rangle$ be the ordering of P given by $\langle p_0, p_1 \rangle$ is extended by $\langle q_0, q_1 \rangle$ and for i in $\{0,1\}$ and e less than n,

$$(3.4) \qquad (\forall x)\,[\langle e, x \rangle \in \text{domain}(q_i) - \text{domain}(p_i) \implies q_i(\langle e, x \rangle) = F_e(x)]\,.$$

In P_n, an extension must code $F_0, \ldots, F_{n-1}$ into the first n columns of K_0 and K_1.

Let Φ_n, Ψ_n be a list of all of the pairs of PTIME functionals. We build K_0 and K_1 by recursion. During the $s+1$st stage of the recursion, we are given $p(s)$ equal to $\langle p_0(s), p_1(s) \rangle$. Let $\langle p_0(s+1), p_1(s+1) \rangle$ be defined so that

(1) Each coordinate of $\langle p_0(s+1), p_1(s+1) \rangle$ has domain containing all length $s+1$ binary strings.

(2) $\langle p_0(s+1), p_1(s+1) \rangle \leq_{P_{s+1}} \langle p_0(s), p_1(s) \rangle$.

(3) If there is a string x and q extending $p(s)$ in P_{s+1} such that q strongly forces $\Phi_{s+1}(K_0, x) \neq \Psi(K_1, x)$, then $p(s+1)$ is such a condition.

Let K_0 be the union of all of the $p_0(s)$ and K_1 the union of the $p_1(s)$. By (1), K_0 and K_1 are defined on all binary strings. By (2), for each n, except for the strings in the domain of $p_0(n)$ or the domain of $p_1(n)$, $K_0^{(n)}$ and $K_1^{(n)}$ are equal to F_n. Thus, $\mathcal{I} \subseteq (K_0) \wedge (K_1)$.

Suppose Φ_n and Ψ_n are given. There are two cases. In the first case, there are a string x and two extensions q_0 and q_0' of $p_0(n-1)$ that satisfy equation 3.4 for all e less than or equal to n relative to $p_0(n-1)$ and that strongly force incompatible values for $\Phi_n(K_0, x)$. We could take any extension q_1 of $p_1(n-1)$ that satisfies 3.4 for all e less than or equal to n and strongly decides $\Psi_n(K_1, x)$; one of q_0 or q_0' would give a value for $\Phi_n(K_0, x)$

that is incompatible with the value forced by q_1. By (3), $p(s+1)$ would be chosen to strongly force $\Phi_n(K_0)$ to be unequal to $\Psi_n(K_1)$.

For the second case, suppose for every x and every two extensions q_0 and q_0' of $p_0(n-1)$ that satisfy 3.4 as above and strongly decide $\Phi_n(K_0, x)$, $\Phi_n(q_0, x)$ is equal to $\Phi_n(q_0', x)$. Then, for all x, $\Phi_n(K_0, x)$ is decided by $p(n-1)$ in P_n. Every $p(s)$ extends $p(n-1)$ in P_n, so if $p(s)$ strongly forces a value for $\Phi_n(K_0)$ then that value is equal to the one forced by $p(n-1)$. Thus, $\Phi_n(K_0, x)$ can be computed by evaluating $\Phi_n(-, x)$ relative to the any extension of $p_0(n-1)$ in P_n that strongly forces a value. In particular, the conditions that extend $p(n-1)$ by coding $F_0, \ldots, F_n$ on the first $n+1$ columns and being 0 elsewhere can be used to evaluate $\Phi_n(K_0)$. Since, this set of conditions is PTIME relative to $F_0 \oplus \ldots \oplus F_n$, $\Phi_n(K_0)$ is PTIME relative to $F_0 \oplus a \ldots \oplus F_n$.

Hence, if $\Phi_n(K_0)$ is equal to $\Psi_n(K_1)$ then their common value is in $\mathcal{I}$. ◇

Now, we turn to the effective version of Theorem 3.3.

3.5. DEFINITION. Let $\langle R_e \mid e \in N \rangle$ be a uniformly recursive list of the partial recursive subsets of $\{0,1\}^*$. An ideal $\mathcal{I}$ in REC has a *recursive presentation* if there is a recursive function f such that

$$\mathcal{I} = \left\{ R_{f(e)} \,\middle|\, e \in N \right\}.$$

(In particular, each $R_{f(e)}$ must be a total recursive set.)

3.6. THEOREM. See also [**1**, Ambos-Spies] *Let $\mathcal{I}$ be an ideal in $\langle REC, \leq_p \rangle$. There are recursive sets K_0 and K_1 such that $\mathcal{I}$ is equal to $(K_0) \wedge (K_1)$ if and only if $\mathcal{I}$ has a recursive presentation.*

PROOF: First, let K_0 and K_1 be recursive sets. The set of reals that are PTIME in K_0 and K_1 can be recursively presented as follows. Let Φ_n and Ψ_n be a recursive list of all of the PTIME functionals. Define X_n by

$$X_n(x) = \begin{cases} \Phi_n(K_0, x), & \text{if } (\forall y)[|y| \leq |x| \implies \Phi_n(y) = \Psi_n(y)] \\ 0, & \text{otherwise.} \end{cases}$$

The proof of the other direction of Theorem 3.6 is the effective version of the proof of Theorem 3.4. Instead of taking the diagonal steps directly, we use strategies to meet the diagonal condition if possible.

Let Φ_n and Ψ_n be a PTIME listing of all pairs of PTIME functionals. The diagonal requirements are, for each n, I(n)

$$\Phi_n(K_0) = \Psi_n(K_1) = Z \implies Z \in \mathcal{I}.$$

We define a strategy for I(n) assuming that the total effect of the higher priority strategies is to impose P_n for the sets $R_{f(0)}, \ldots, R_{f(n-1)}$. The strategies in this section are easier to work with than in the general case. During stage $s+1$, I(n) is defined from inputs $s+1$, $p(s)$ and $in(s+1)$.

(1) **Transition:** If there is an extension q of $p(s)$ and a string x with $|x|$ less than $in(s+1)$ such that $q \leq_{P_n} p(s)$ as verified by computations of $R_{f(0)}, \ldots, R_{f(n-1)}$ such that

$$q \Vdash^* \Phi_n(q_0, x) \neq \Psi_n(q_1, x)$$

go to (2).

We look at all strings of length less than $in(s+1)$ for a possible diagonal step. Note, this involves the evaluation of the first n recursive sets on no more than the strings of length bounded by the maximum of $u_{\Phi_n}(in(s+1))$ and $u_{\Psi_n}(in(s+1))$. Since f only gives indices for total recursive predicates, either we find a condition strongly forcing an inequality or we check all relevant conditions.

Environment: No restrictions.

(2) **Environment:** Require

$$p(s+1) \Vdash^* \Phi_n(K_0) \neq \Psi_n(K_1).$$

Use the diagonal condition found in (1).

3.7. LEMMA. *Suppose that $\mathcal{C}$ is a construction that is eventually coherent relative to I(n) and eventually respects I(n). The requirement I(n) is satisfied by the sets produced.*

PROOF: If $\Phi_n(K_0)$ is not equal to $\Psi_n(K_1)$ then the requirement is satisfied. Assume otherwise.

Let s_0 be the least stage such that for all $s+1$ greater than or equal to s_0, $\mathcal{C}$ respects I(n) and is coherent with respect to I(n) during stage $s+1$. I(n) cannot reach state (2) after stage s_0 or we would be in the first case. Given x, we compute $\Phi_n(K_0, x)$ from

$\mathcal{R}_{f(0)} \oplus \cdots \oplus R_{f(n-1)}$ and a table of values of $\Phi_n(K_0)$ at arguments of length less than or equal to $in(s_0)$ as follows.

If $|x| \leq in(s_0)$,

Then look up $\Phi_n(K_0, x) = answer$ in the table of values;

Else:

Compute the least s such that $in(s+1) > |x|$;

Compute $p_0(s)$;

Simulate the computation of $\Phi_n(K_0, x) = answer$ responding to a query y by:

Cases:

$p_0(s)(y)$, if y is in the domain of $p_0(s)$;

$R_{f(e)}(z)$, if y is of the form $\langle e, z \rangle$ and $e < n$;

0, otherwise;

End if;

Output *answer*;

End.

The same argument as in the proof of Theorem 3.4 applies to show that the answer computed above is equal to $\Phi_n(K_0)$. The coherence of the construction implies that the computation of s in the first line and the later computation of $p_0(s)$ are both PTIME in $|x|$. Thus, the entire algorithm is PTIME in $R_{f(0)} \oplus \cdots \oplus R_{f(n-1)}$. ◊

The coding requirements II(n) have the form, for all n,

$$R_{f(0)} \oplus \cdots \oplus R_{f(n-1)} \in (K_0) \wedge (K_1).$$

Let P_n be defined as in the proof of Theorem 3.4, for the sets $R_{f(0)}, \ldots, R_{f(n-1)}$. Our strategy at stage $s+1$, is to impose the environment P_n given by the sets $R_{f(0)}, \ldots, R_{f(n-1)}$. Clearly, this strategy ensures the satisfaction of its requirement.

3.8. LEMMA. *The sequence of strategies*

$$\vec{\mathcal{S}} = I(0), II(0), I(1), II(1), \ldots$$

is compatible.

PROOF: We identify a strategy with the singleton family it generates. The coding strategies operate on different columns and the diagonal strategies only change the environment to impose a finite constraint when they change state.

The method to find a condition in the common environment imposed by an initial segment of strategies is to first assume that no diagonal condition will be found. Given $\overrightarrow{S}$, an initial segment of $\overrightarrow{\mathcal{S}}$, an integer ℓ and construction respecting $\overrightarrow{S}$ wherein the elements of $\overrightarrow{S}$ have achieved states $\vec{\sigma}$, we extend $p(s)$ to q as follows. Let P_n be the coding environment imposed by the type II elements of $\overrightarrow{S}$. Let q extend $p(s)$ by being 0 on every string of length less than or equal to ℓ, that is not involved in the coding strategies of type II and as determined by equation 3.4, elsewhere. If, a diagonalizing condition is found during the subsequent generation of the sequence $\overrightarrow{m}(s+1)$, use the same method to extend the least diagonalizing condition relative to the initial segment of $\overrightarrow{S}$ before the one that diagonalized. ◊

We can now complete the proof of Theorem 3.6. Since each strategy of type I(n) or II(n) can change state at most once, from (1) to (2), they change state finitely often. Thus, by Lemma 3.8, Theorem 2.10 applies: there is coherent recursive construction of sets K_0 and K_1 that is eventually respects all of the strategies I(n) and II(n). By Lemma 3.7 and the fact that the type II strategies satisfy their requirements, the sets produced by this construction satisfy the requirements of Theorem 3.6. ◊

§4. INTERPRETATIONS OF ARITHMETIC

Representing partially ordered sets. Let $\mathcal{D}$ be an upper semi-lattice with join $\oplus$.

4.1. DEFINITION. Let U be an element of $\mathcal{D}$ and I be a subset of $\mathcal{D}$.

(1) I is *independent over U in $\mathcal{D}$* if for every finite set $F = \{X_0, \ldots, X_n\}$ contained in I and every X in I, if X is not an element of F, then $X \not\leq_p X_0 \oplus \ldots \oplus X_n \oplus U$.

(2) I is *strongly independent over U in $\mathcal{D}$* if for every X in I there is a Y in $\mathcal{D}$ such that

$$(\forall Z \in I)[Z = X \iff Z \not\leq_{\mathcal{D}} U \oplus Y].$$

If I is strongly independent over U then I is independent over U. We introduce strong independence because it is a first order property.

4.2. DEFINITION. Let $\preceq$ be a partial ordering of N, the nonnegative integers. The parameters A, B, C, K_0, K_1, L and U (from $\mathcal{D}$) *code* $\preceq$ in $\mathcal{D}$ if and only if there is a set of degrees $\mathcal{G}$ equal to $\{G_i \mid i \in N\}$ such that the following conditions are satisfied in $\mathcal{D}$.

(1) For each Y in $(\mathcal{G})$, either
   (a) $(A) \wedge (B \oplus Y) \not\subseteq (C \oplus U \oplus Y)$ or
   (b) There is a G_i in $\mathcal{G}$ such that $G_i \leq_{\mathcal{D}} U \oplus Y$.

(2) For each G_i in $\mathcal{G}$, $(A) \wedge (B \oplus G_i) \subseteq (C \oplus U \oplus G_i)$.

(3) $\mathcal{G}$ is strongly independent over U.

(4) For all i and j in N, $G_j \leq_{\mathcal{D}} G_i \oplus L \oplus U$ if and only if $j \preceq i$.

(5) $(K_0) \wedge (K_1)$ is equal to $(\mathcal{G})$.

4.3. PROPOSITION. *Suppose that there are parameters as in Definition 4.2 coding* $\preceq$ *in* $\mathcal{D}$. *There is a recursive translation* $\varphi \mapsto \varphi^*$ *such that for any sentence* φ

$$\langle N, \preceq \rangle \models \varphi \iff \mathcal{D} \models \varphi^*(A, B, C, K_0, K_1, L, U).$$

PROOF: It is enough to show that there a relation on elements of $\mathcal{D}$ that is an isomorphic copy of $\langle N, \preceq \rangle$ and uniformly definable from A, B, C, K_0, K_1, L and U.

By (5), $(\mathcal{G})$ is definable in $\mathcal{D}$. By (4), the set

$$\mathcal{G} \oplus U = \left\{ G_i \oplus U \,\middle|\, i \in N \right\}$$

is strongly independent in $\mathcal{D}$ and so any two of its elements are incomparable. This together with (1) and (2) shows that $\mathcal{G} \oplus U$ is definable as the set of minimal elements of

$$\left\{ Y \oplus U \,\middle|\, \begin{array}{c} Y \in (K_0) \wedge (K_1) \;\& \\ (\forall Z)\,[Z \in (A) \wedge (B \oplus Y) \implies Z \in (C \oplus U \oplus Y)] \end{array} \right\}.$$

Now consider the partial ordering of $\mathcal{G} \oplus U$ given by $G_i \oplus U \leq G_j \oplus U$ if and only if $G_i \oplus U \leq_{\mathcal{D}} G_j \oplus L \oplus U$. This is a definable partial ordering in $\mathcal{D}$ of a definable subset of $\mathcal{D}$; by (4), $\langle \mathcal{G} \oplus U, \leq \rangle$ is isomorphic to $\langle N, \preceq \rangle$. $\Diamond$

Representing models of arithmetic. We begin by fixing a partial ordering P_N that represents the natural numbers with the successor function. We will present P_N in terms of generators and relations. Let $\leq_N$ denote the ordering in P_N.

(1) The minimal elements of P_N form a set $\{m_i \mid i \in N\}$.

(2) For each i and j, there is a unique pair $c_+(i,j)$ and $d_+(i,j)$ such that $c_+(i,j) \geq_N m_i, m_j$ and $d_+(i,j) \geq_N c_+(i,j), m_{i+j}$.

(3) For each i and j, there is a unique triple $c_\times(i,j)$, $d_\times(i,j)$ and $e_\times(i,j)$ such that $c_\times(i,j) \geq_N m_i, m_j$, $d_\times(i,j) \geq_N c_\times(i,j), m_{i\times j}$ and $e_\times(i,j) \geq d_\times(i,j)$.

The only elements of P_N are the ones mentioned above. The only instances of $\leq_N$ comparability are those needed to obtain a transitive relation satisfying the above clauses.

The integers of P_N are endowed the with the structure of arithmetic in a way that is P_N-definable . For example, in P_N define $m_i +_{P_N} m_j = m_k$ if and only if there exist x and y in P_N such that $x \geq_N m_i, m_j$; $y \geq x, m_k$; and y is maximal in P_N. The following proposition is built into our definition of P_N.

4.4. PROPOSITION. (1) *There is a recursive partial ordering $\preceq$ of N that is isomorphic to P_N. In addition, in the coding by $\preceq$, the set of integers, addition and multiplication are also recursive.*

(2) *There is an interpretation of the first ordered theory of N in the first order theory of P_N.*

To avoid making a constant translation between the language of arithmetic and the language of partially ordered sets we will speak of P_N as if it were N. We will also identify P_N with the recursive partial ordering of N referred to in Proposition 4.4. We let $+_N$ and $\times_N$ denote the operations of addition and multiplication on the integers of P_N. Similarly, when a partial order P has enough of the properties of P_N to define a set of integers with addition and multiplication as above, we will call P an interpretation of the language of arithmetic.

One of the first theorems about the axiomatics of N is that N has a second order characterization. There is a finite set of first order sentences P^- such that if M is a model of P^- and every subset of M has a least element, then M is isomorphic to N. The second condition is usually expressed by saying that M satisfies full induction. Suppose that M is a model of P^-. Let 1_M be the successor of the minimal element of M. Full induction for M is equivalent to the following condition: for all m in M, either m is equal to 0_M or there is a k in N such that

$$m = \underbrace{1_M +_M \cdots +_M 1_M}_{k \text{ times}}. \tag{4.5.}$$

Say that m is a *standard* element of M if either m is 0_M or there is a k in N such that Equation 4.5 holds. The standard part of M is the set of its standard elements. M is isomorphic to N if and only if M is equal to its standard part.

4.6. DEFINITION. Suppose that M is a partially ordered set that interprets the language of arithmetic. Say that M is a *model of arithmetic* if M satisfies P^-. M is a *nonstandard model* if M is not isomorphic to P_N, or equivalently, M has a nonstandard element.

4.7. PROPOSITION. *Suppose that $\mathcal{D}$ is an upper semi-lattice such that there is a $\mathcal{D}$-definable predicate S such that*

(1) *$\vec{p} \in S$ implies that $\vec{p}$ is a sequence of parameters coding a model of arithmetic (as in Definition 4.2) wherein the set of integers is independent.*

(2) *If $M(\vec{p})$ is the model coded by some $\vec{p} \in S$, then there is pair K_0 and K_1 such that for every integer m in $M(\vec{p})$,*

$$m \text{ represents a standard element of } M(\vec{p}) \iff m \in (K_0) \wedge (K_1).$$

(3) *There is at least one element $\vec{p}$ of S that codes a partial ordering isomorphic to P_N.*

Then, there is an interpretation of the first order theory of arithmetic in the first order theory of $\mathcal{D}$.

PROOF: By (1) and (2), we can use S to define a collection of codes for standard models of arithmetic. By (3), this collection is not empty. Given a sentence φ in the language of arithmetic, let $\varphi^*(\vec{p})$ be the sentence in the parameters $\vec{p}$ that is equivalent to saying that $\vec{p}$ codes a standard model of arithmetic and φ is satisfied in the partial order coded by $\vec{p}$. Then,

$$N \models \varphi \iff \mathcal{D} \models (\exists \vec{p})\varphi^*(\vec{p}).$$

This gives an interpretation of the theory of N in the theory of $\mathcal{D}$. ◊

4.8. PROPOSITION. *Work with the same notation and assumptions as in Proposition 4.7. Suppose further, for any model coded in $\mathcal{D}$ and any subset S of the integers of that model there is a pair H_0 and H_1 such that $(H_0) \wedge (H_1)$ is equal to (S). Then, there is an interpretation of the second order theory of arithmetic in the first order theory of $\mathcal{D}$.*

PROOF: As in Proposition 4.7, we obtain a definable collection of coded standard models of arithmetic. By clause (1), the integers of a coded model form an independent set. Thus,

every subset S of the integers in a coded model is determined by the ideal it generates. Using the additional assumption, these are determined by exact pairs. Thus we can interpret second order variables over a coded model in a way that is first order in $\mathcal{D}$ by using exact pairs. $\Diamond$

We will now state the technical results that apply to $\langle PTIME, \leq_p \rangle$ and $\langle \mathcal{R}, \leq_p \rangle$.

4.9. THEOREM. *There are recursive sets A, B, C, K_0, K_1, L, P_0, Q_0, P_1, Q_1, U and a sequence $\mathcal{G} = \{G_i \mid i \in N\}$ of recursive sets with the following properties.*

(1) *A, B, C, K_0, K_1, L, and U code a partial ordering of $\mathcal{G} \oplus U$ that is isomorphic to P_N.*

(2) *For all m, an even integer in the sense of P_N,*

$$(G_m \oplus P_0 \oplus U) \wedge (Q_0 \oplus U) = (G_{m'} \oplus U),$$

where m' is the successor of m in P_N. Similarly, if m is an odd integer in P_N,

$$(G_m \oplus P_1 \oplus U) \wedge (Q_1 \oplus U) = (G_{m'} \oplus U).$$

Clause (2) and its use in the following proof were directly adapted from [**9**, Shore]. We will present the proof of Theorem 4.9 in sections §5 to §7.

4.10. THEOREM. (1) *There is an interpretation of the first order theory of arithmetic in the first order theory of $\langle REC, \leq_p \rangle$.*

(2) *There is an interpretation of the theory of second order arithmetic in the first order theory of $\langle \mathcal{R}, \leq_p \rangle$.*

PROOF: (Assuming Theorem 4.9.) In the following, when we will say that a predicate R is definable in either of $\langle REC, \leq_p \rangle$ or $\langle \mathcal{R}, \leq_p \rangle$, we will mean that there are finitely many parameters $\vec{p}$ and a first order formula φ so that R is equal to the set of solutions to $\varphi(\vec{p}, -)$ in the structure. If the same definition works in both structures, we will merely say that R is definable.

We begin with the first claim. Working in $\langle REC, \leq_p \rangle$, we wish to show that there is a definable nonempty collection of codes of standard models of arithmetic. We simultaneously present the definition and prove that it is correct.

In stating φ, we begin with $\vec{p}$, a sequence A, B, C, K_0, K_1, L, P_0, Q_0, P_1, Q_1 and U of recursive sets. Let $\mathcal{G} \oplus U$ be the set of minimal elements of

$$\left\{ Y \oplus U \;\middle|\; \begin{array}{c} Y \in (K_0) \wedge (K_1)\ \& \\ (\forall Z)\,[Z \in (A) \wedge (B \oplus Y) \implies Z \in (C \oplus U \oplus Y)] \end{array} \right\}$$

We require that $\mathcal{G} \oplus U$ be strongly independent (see Definition 4.1). Next, we require that $\vec{p}$ code a partial ordering of $\mathcal{G} + U$, in the sense of Definition 4.2, that satisfies P^-. Further, the addition operation of the coded model must satisfy condition (2) in Theorem 4.9 with respect to P_0, Q_0, P_1 and Q_1.

Lastly, we require for all H_0 and H_1, if $(H_0) \wedge (H_1)$ is closed under the successor operation of the coded model then every integer in the coded model is an element of $(H_0) \wedge (H_1)$. These requirements form a first order property of $\vec{p}$.

By Theorem 4.9, there is a $\vec{p}$ that satisfies the requirements.

Let $\vec{p}$ be fixed to satisfy the above requirements. Let M be the model of P^- coded by $\vec{p}$. Recall that M is represented as a definable partial ordering of $\mathcal{G} \oplus U$. Consider the set $\mathcal{I}$ defined by $X \in I$ if and only if there is a finite sequence $Z_0 \ldots, Z_n$ such that Z_0 is the 0th element of M, for each even (odd) m less than n, Z_{m+1} is PTIME in both $P_0 \oplus Z_m$ and Q_0 ($P_1 \oplus Z_m$ and Q_1, respectively). As in the proof of Theorem 3.6, $\mathcal{I}$ has a recursive presentation: effectively list all ways of producing such finite sequences and sets computed from their last element; if a sequence of pairs of PTIME reductions is seen not to compute a sequence of Z_i's, the set we were computing from the final Z_n is converted into a finite set. By Theorem 3.6, there are H_0 and H_1 such that $(H_0) \wedge (H_1)$ is equal to $\mathcal{I}$.

By condition (2) in Theorem 4.9, $\mathcal{I}$ is the ideal generated by the elements of $\mathcal{G} \oplus U$ that are standard elements in M. By the final condition on $\vec{p}$, $\mathcal{I}$ must include all of the integers of M. Since $\mathcal{G} \oplus U$ is strongly independent, $\mathcal{I}$ does not include any elements of $\mathcal{G} \oplus U$ other than the standard integers. Thus, every integer in M is standard and so M is isomorphic to P_N.

Now we can invoke Proposition 4.7 to conclude the first claim from the ability to define a nonempty collection of standard models.

The second claim is much easier to see. Work in $\langle \mathcal{R}, \leq_p \rangle$. Consider the set of sequences $\vec{p}$ that code models of P^- in which the set $\mathcal{G} \oplus U$ is strongly independent. By theorem 4.9, this set is not empty. We can recognize the sequences that code standard models

as above using Theorem 3.3 to provide an exact pair above the ideal generated by the standard integers. Using Theorem 3.3 again to provide an exact pair for every subset of the integers. We can invoke Proposition 4.8 to conclude (2). ◊

§5. Requirements

In this section, we introduce the requirements associated with the proof of Theorem 4.9 and describe the qualitative features of the sets to be constructed.

We are required to produce recursive sets A, B, C, K_0, K_1, L, P_0, Q_0, P_1, Q_1, U and a recursive sequence of sets $\mathcal{G} = \{G_i \mid i \in N\}$. We will simultaneously and uniformly construct all of these sets, except for K_0 and K_1, as subsets of $\{0,1\}^*$. We will apply Theorem 3.6 to conclude the existence of K_0 and K_1.

The requirements are naturally divided into the following parameterized families.

I(Φ, $\overrightarrow{G}$). The first type of requirement is syntactically the most complicated. We write it as a collection of related requirements. Given Φ and a finite sequence $\overrightarrow{G}$ from $\mathcal{G}$, let Y denote $\Phi(\overrightarrow{G})$.

I(Φ, $\overrightarrow{G}$, global). We build Γ and Δ to satisfy the global requirement

$$\Gamma(A) = \Delta(B \oplus Y).$$

I(Φ, $\overrightarrow{G}$, Ψ). Further, for each Ψ, we satisfy one of the following. Either,

$$\Psi(C \oplus U \oplus Y) \neq \Gamma(A)$$

or for some G_i in $\overrightarrow{G}$, we build Π and satisfy

$$\Pi(U \oplus Y) = G_i.$$

II(Θ, Ω, i). Given Θ and Ω, we build a functional Σ so that

$$\Theta(A) = \Omega(B \oplus G_i) = Z \implies Z = \Sigma(C \oplus U \oplus G_i).$$

III(Θ, j). Given Θ, i and j, we satisfy

$$\Theta(\oplus_{i \neq j} G_i \oplus U) \neq G_j.$$

IV(i, j, coding). For each i and j such that $j \leq_{P_N} i$, we build Λ so that

$$\Lambda(G_i \oplus L \oplus U) = G_j.$$

IV(Θ, i, j). For each Θ, i and j so that $j \not\leq_{P_N} i$, we satisfy

$$\Theta(G_i \oplus L \oplus U) \neq G_j.$$

Let m be an even integer in the sense of P_N with successor m'.

V(m, coding). We build Γ and Δ to satisfy

$$\Gamma(P_0 \oplus G_m) = \Delta(Q_0) = G_{m'}$$

V(m, Φ, Ψ). Given Φ and Ψ, we build Δ so that

$$\Phi(G_m \oplus P_0 \oplus U) = \Psi(Q_0 \oplus U) = Z \implies \Delta(G_{m'} \oplus U) = Z.$$

Define **V(m, coding)** and **V(m, Φ, Ψ)** for m an odd integer in P_N similarly, replacing 0 by 1.

We leave it to the reader to show that any sets satisfying the above requirements satisfy Theorem 4.9. Requirements I-IV and VI correspond to the conditions appearing in the definition of coding a partial ordering (see Definition 4.2). Requirement V corresponds to the second clause in Theorem 4.9.

The roles of the parameters. The sets play well defined roles in the construction.

U is a very thinly distributed set, all of whose elements are strings of 0's. Almost every action will take place relative to a string in U. For example, the elements of $\mathcal{G}$ will be subsets of U. When we establish an inequality between a set and the value of some functional, it will invariably occur at an argument in U.

A is an infinite join of sets. The columns of A are the sets coded into the meet of (A) with $(B \oplus Y)$ for the sake of requirements of type I.

B is also an infinite join of sets. Each column of B is a set of strings coding quadruples $\langle x, i, pos, neg \rangle$: x is a string; i is either 0 or 1; pos and neg are codes for disjoint subsets of the set of strings of length less than or equal to $\log(\log(|x|))/2$. Working on the eth

requirement, we think of $\langle x, i, pos, neg\rangle \in B^{(e)}$ as providing a computation relative to $B \oplus Y$. Namely, if $\langle x, i, pos, neg\rangle \in B^{(e)}$, *pos* codes a subset of Y and *neg* codes a subset of the complement of Y then the answer at x is i. We define a (linear time) Turing computation relative to $B \oplus Y$ as follows. If there is a computation $\langle x, i, pos, neg\rangle \in B^{(e)}$ relevant to Y then output answer i; otherwise, output answer 0.

The columns of C contain information about the actions of strategies associated with requirements of type II. For each element x of U and each strategy, we have the option of putting one of three flags on x into the pertinent column of C. The flags will be used to simulate a computation relative to A using $C \oplus U \oplus G_i$.

The remaining sets are as independent as is possible given the coding requirements.

§6. Strategies

In this section, we specify the families of strategies used to satisfy the requirements. As usual, in the specification of a strategy, we are allowed to use inputs $C\restriction s$ and $in\restriction s+1$. We will describe the stage $s+1$ behavior of the strategies in terms of these using an auxiliary function $gap(s)$, which in the final construction, will have a PTIME graph. The function *gap* is used to introduce a long blank interval between the locations of strings used during stage s and those used during stage $s+1$.

The initial environment. The first strategy is not associated with any particular requirement. It forces the sets produced to have the qualitative properties described in section §5. During stage $s+1$, it imposes the environment $E_0(s+1)$. $E_0(s+1)$ is defined by specifying the properties of the conditions q that belong to it. Note, that the definition of $E_0(s+1)$ depends only on $s+1$, $p(s)$ and $gap(s)$.

(1) *Basics.* The predicates A, B, C, L, P_0, Q_0, P_1, Q_1 and U are named in q. The remaining predicates named in q are contained in $\{G_i \mid i \leq s+1\}$.

(2) *Restrictions on A.* All of the elements of $q(A)$ are of the form $\langle e, z\rangle$ where e is less than or equal to $s+1$ and z is an element of $q(U)$. There is at most one element in $q(A) - p(s)(A)$. *The columns of A will be subsets of U.*

(3) *Restrictions on B.* All of the elements of $q(B)$ are of the form $\langle e, \langle x, i, pos, neg\rangle\rangle$: e is less than or equal to $s+1$; x is a string; i is either 0 or 1; *pos* and *neg* are codes for disjoint subsets of the set of strings of length less than $\log(\log(|x|))/2$.

(4) *Restrictions on C.* All of the elements of $q(C)$ are of the form $\langle e, z, 1\rangle$ or $\langle e, z, 2, i\rangle$ where e is less than or equal to $s+1$, z is an element of U and i is either 0 or 1. These are called *flags* for z. There is at most one element in $q(C) - p(s)(C)$.

(5) *Restrictions on U.* The only elements of $q(U)$ are strings of 0's. If $q(U)$ is defined on 0^m, then for all n less than m, $q(U)$ is defined on 0^n. The shortest element of $q(U) - p(s)(U)$ has length greater than $gap(s)$.

(6) *Restrictions on G_k.* For all strings x, if $q(G_k)(x) = 1$ then $q(U)(x) = 1$. If G_k is named in q but not in $p(s)$, then $q(G_k)(z)$ is equal to 0, for every string z of length less than or equal to $\ell_{p(s)}(U)$. *Thus, the elements of $\mathcal{G}$ will be subsets of U.*

(7) *Restrictions on ℓ_q.* There is an x in $q(U)$ such that for each X named in q other than U, $\ell_q(X)$ is less than $|x|$. For all G_i and G_j named in q, $\ell_q(G_i) = \ell_q(G_j)$. For all G_k named in q, for all x and e, if any of $q(A)(\langle e, x\rangle)$, $q(B)(\langle e, \langle x, i, pos, neg\rangle\rangle)$, $q(C)(\langle e, x, 1\rangle)$, or $q(C)(\langle e, x, 2, i\rangle)$ is defined, then $q(G_k)(x)$ is defined.

6.1. DEFINITION. (1) An environment E, depending on $s+1$, $p(s)$ and $in(s+1)$ is *U-free* over $f : N \to N$ if for any q in E and U' extending $q(U)$ such that

$$(\forall n, m \in N)\left[\begin{array}{c}(0^m \in U' - q(U)\ \&\ 0^n \in U'\ \&\ m > n)\\ \Longrightarrow\ (m > gap(s)\ \&\ m > f(n))\end{array}\right]$$

there is an r extending $q * U'$ such that r is in E.

(2) E imposes the condition that U *dominate* f when, for all q in E, if y is an element of $q(U) - p(s)(U)$, z is in $q(U)$ and $|z| < |y|$, then $|y| > f(|z|)$.

(3) E is *$\mathcal{G}$-free* if, for any q in E and any $\vec{G}'$ extending $q(\vec{G})$ such that, for all G'_i in $\vec{G}'$, if $G'_i(x) = 1$ then $q(U)(x) = 1$, there exists an r extending q in E such that $r(\vec{G})$ extends $\vec{G}'$.

(4) E is *determined by f on $\mathcal{G}$ and U* if E imposes the constraint that U dominate f, E is U-free over f and E is $\mathcal{G}$-free.

If E imposes the constraint that U dominate f and E is U-free over f, then the only constraint on U is that it respect the *gap* condition of $E_0(s+1)$, clause (5), and that its enumerating function grow faster than the f. All of our strategies will produce environments of this sort.

I(Φ, $\vec{G}$)-strategies. The strategies associated with I(Φ, $\vec{G}$) fall into two types as did

the clauses in the statement of the requirement. For the discussion of these strategies, let Y denote $\Phi(\overrightarrow{G})$.

I(Φ, $\overrightarrow{G}$, global). Suppose that this strategy is given index e. In the global strategy, we define a functional Δ in PTIME and set the environment so that $A^{(e)}$ will equal $\Delta(B \oplus Y)$. The strategy has only one state. Given input $\mathcal{C} \restriction s$ and $in(s+1)$, its imposed environment consists of the conditions q with the following properties.

(1) **Environment:** We extend the constraint imposed in $E_0(s+1)$ so that, if $\langle x, i, pos, neg\rangle$ is an element of $q(B^{(e)})$, then q strongly decides whether pos is a subset of Y and neg is contained in the complement of Y. If q forces $pos \subseteq Y \ \& \ neg \cap Y = \emptyset$, then $q(A^{(e)})(x) = i$. Further, if $A^{(e)}(x) = 1$, then there is such a quadruple in $q(B^{(e)})$. Lastly, $\langle e, x\rangle$ is in the domain of $q(A)$ if and only if for all pos and neg as above, $\langle x, 0, pos, neg\rangle$ and $\langle x, 1, pos, neg\rangle$ are in the domain of $q(B^{(e)})$ if and only if there is one such quadruple in the domain of $q(B^{(e)})$.

By the first clauses, if our functional $\Delta(B \oplus Y)$ is defined at z by an quadruple in B, then its value is equal to $A^{(e)}(z)$. If no element of B sets the value, then $A^{(e)}$ has the default value 0. The next clause ensures that when it applies, the default value is correct. Lastly, $A^{(e)}$ is decided at x if and only if the evaluation of $\Delta(B \oplus Y)$ at x is decided.

6.2. LEMMA. *If $\mathcal{C}$ is a coherent construction that eventually respects I(Φ, $\overrightarrow{G}$, global) then requirement I(Φ, $\overrightarrow{G}$, global) is satisfied by the sets produced.*

PROOF: The functional $\Delta(B \oplus Y)$ is defined as follows. For argument x, if there are pos and neg subsets of the set of strings of length less than or equal to $\log(\log(|x|))/2$ and i either 0 or 1 such that $\langle x, i, pos, neg, \rangle \in B^{(e)}$, $pos \subset Y$ and $neg \cap Y = \emptyset$ then output answer i. Otherwise, output answer 0.

Δ is a Turing functional. During every stage, I(Φ, $\overrightarrow{G}$, global) imposes the condition that $A^{(e)} = \Delta(B \oplus Y)$, hence they are equal in the limit. We need only check that Δ belongs to PTIME. This follows from the elementary calculation showing that there are on the order of $|x|$ many quadruples to be checked in the evaluation of $\Delta(B \oplus Y, x)$. The exact coding mechanism is not important. We will only assume that the coding is fixed so that the set of codes for quadruples with first coordinate x is uniformly polynomial in x. $\Diamond$

I(Φ, $\overrightarrow{G}$, Ψ). Using I(Φ, $\overrightarrow{G}$, global), we force $A^{(e)}$ belong to $(A)\wedge(B\oplus Y)$. The current strategy will ensure that either $A^{(e)}$ is not equal to $\Psi(C\oplus U\oplus Y)$ or there is an element G_i of $\overrightarrow{G}$ such that $G_i \leq_p Y$.

Given u_Φ and u_Ψ, increasing polynomials giving time bounds on the computations of Φ and Ψ, let $u_\Phi \circ u_\Psi$ be their composition.

The strategy for I(Φ, $\overrightarrow{G}$, Ψ) rides on the difference between forcing and strong forcing. We begin with some preliminary definitions and lemmas.

6.3. DEFINITION. Let Ψ be fixed in this definition. Let E be an environment extending $E_0(s+1)$ that is determined by f on $\mathcal{G}$ and U. Let p be an element of E and x be in $p(U)$ such that all X named in p other than U, $\ell_p(X) < |x|$. Further, suppose that for all y in $p(U)$ either $|y| \leq |x|$ or $u_\Phi \circ u_\Psi(|x|) < |y|$. Say that C_0 is compatible with p if there is a $q <_E p$ such that C_0 is contained in $q(C)$.

(1) Y is *nonsplitting at x over p* if there is a q extending p in E such that
   (i) For all G_i named in q, $\ell_q(G_i) < |x|$.
   (ii) If C_0 is compatible with p in E, then $Y\restriction \psi(C_0\oplus U\oplus Y, x)$ (Y restricted to the $C_0\oplus U\oplus Y$ computation of Ψ at x) is decided by q in E.
   (iii) For all X not in $\overrightarrow{G}$, $q(X) = p(X)$.

(2) Y is *G_i-splitting at x over p* if
   (i) If C_0 is compatible with p then for any $q <_E p$, the values forced by q in E for $Y\restriction \psi(C_0\oplus U\oplus Y, x)$ are forced by p and $q(G_i)\restriction |x|$.
   (ii) If q and r are extensions of p in E that are incompatible with regard to G_i at x, then there is a C_0, compatible with p, such that q and r force incompatible values for $Y\restriction \psi(C_0\oplus U\oplus Y, x)$.

(3) Y is *amorphous at ℓ over p* if there is a q extending p in E and strongly forcing a value for $Y(z)$ at every string z with $|z| \leq \ell$, such that for all G_i mentioned in p, there is an extension r of $p * q(G_i) * q(U)$ and a z of length less than ℓ such that r forces a value for $Y(z)$ that is incompatible with the one forced by q.

6.4. LEMMA. *Suppose that f dominates $u_\Phi \circ u_\Psi$ point wise. Work in an environment E extending $E_0(s+1)$ that is determined by f on $\mathcal{G}$ and U. Suppose that $p <_E p(s)$, x is in $p(U) - p(s)(U)$ and for all G_k named in p, x is the shortest element of $p(U)$ not in*

the domain of G_k. One of the following three conditions must hold.

(1) *Y is nonsplitting at x over p.*

(2) *There is a $G_k \in \overrightarrow{G}$ such that Y is G_k-splitting at x over p.*

(3) *Y is amorphous at $u_\Psi(|x|)$ over p.*

PROOF: First, since E imposes the constraint that U dominate $u_\Phi \circ u_\Psi$ and E extends $E_0(s+1)$, $Y \upharpoonright \{0,1\}^{\leq u_\Psi(|x|)}$ is decided by $\overrightarrow{G} \upharpoonright \{0,1\}^{\leq |x|}$ in E. By the choice of x, Y will be decided over p by the values of the elements of $\overrightarrow{G}$ at x. Since E is $\mathcal{G}$-free, any Boolean combination of values at x is compatible with p in E. We work in E below.

If Y is nonsplitting at x over p, the lemma is true. Assume otherwise.

For each k, let $p(G_k) * (G_k(x) = i)$ be the extension of $p(G_k)$ that has value i at x. Let $p * (\overrightarrow{G}(x) = 0)$ be the extension of p that excludes x from entering any element of $\overrightarrow{G}$. Let $\psi^*(Y,x)$ denote the set of strings that are queried to Y in a computation of $\Psi(C_0 \oplus U \oplus Y, x)$ for some C_0 that is compatible with p. Note that $\psi^*(Y,x)$ is contained in $\{0,1\}^{\leq u_\Psi(|x|)}$. If there is no k such that $p * (G_k(x) = 0)$ decides $Y \upharpoonright \psi^*(Y,x)$ in E then Y is amorphous at $u_\Psi(|x|)$ over p. Assume otherwise; fix k and Y_0 such that

$$p * (G_k(x) = 0) \Vdash Y \upharpoonright \psi(Y,x) = Y_0.$$

If $p * (G_k(x) = 1)$ also decides $Y \upharpoonright \psi^*(Y,x)$, then the value forced by $p * (G_k(x) = 1)$ must be different from Y_0. Otherwise, p would force $Y \upharpoonright \psi^*(Y,x)$ equal to Y_0 and we would be in the nonsplitting case. Thus, if $p * (G_k(x) = 1)$ decides $Y \upharpoonright \psi^*(Y,x)$, Y is G_k-splitting at x over p.

Lastly, suppose that $p * (G_k(x) = 1)$ does not decide $Y \upharpoonright \psi^*(Y,x)$. Let q and r be extensions of $p * (G_k(x) = 1)$ forcing incompatible values for $Y \upharpoonright \psi^*(Y,x)$. As E is determined by a function that dominates $u_\Phi \circ u_\Psi$ on $\mathcal{G}$ and U, deciding G_k through $|x|$ decides G_k through $u_\Phi \circ u_\Psi$. This decides G_k on every string that could be queried in the evaluation of $Y \upharpoonright \psi^*(Y,x)$. We may assume that $q(G_k)$ is equal to $r(G_k)$. One of q or r (say q) must force $Y \upharpoonright \psi^*(Y,x) \neq Y_0$. We can change q by setting G_k equal to 0 at x to force $Y \upharpoonright \psi^*(Y,x) = Y_0$. Thus, if j is not equal to k, then $p * q(G_j)$ does not decide $Y \upharpoonright \psi^*(Y,x)$. Also, we can keep the condition on G_k fixed and change q to r, thereby changing the value forced for $Y \upharpoonright \psi^*(Y,x)$. Thus, $p * q(G_k)$ does not decide $Y \upharpoonright \psi^*(Y,x)$. Hence, q is a witness to Y's being amorphous at x over p. ◊

6.5. LEMMA. *Suppose that E, p and x are given as in the previous lemma, such that Y is G_k-splitting at x over p. Let x_1 be an element of $p(U)$ with $|x| < |x_1|$. At least one of the following conditions holds.*

(1) *Y is nonsplitting at x_1 over p.*

(2) *Y is G_k-splitting over p at x_1.*

(3) *Y is amorphous at $u_\Psi(|x_1|)$ over p.*

PROOF: Fix the notation $\psi^*(Y,x)$ as in the previous lemma. If there is a $q \leq_E p$, with $\ell_q(G_i) < |x_1|$ for all $G_i \in \vec{G}$, that decides $Y \upharpoonright \psi^*(Y,x_1)$ in E, then Y is nonsplitting over p at x_1. This is case (1). Assume otherwise.

Similarly, if for m in $\{0,1\}$, Y is G_k-splitting over $p * (G_k(x) = m)$ at x_1, then case (2) holds. Assume further, that this is not the case. If Y fails to be G_k splitting because of a G'_k extending $p(G_k)$ and deciding G_k no further than $\{0,1\}^{\leq |x_1|-1}$ such that $p * (G'_k) * (G_k(x_1) = 0)$ and $p * (G'_k) * (G_k(x_1) = 1)$ decide $Y \upharpoonright \psi^*(Y,x_1)$ in the same way in E, then Y would be nonsplitting at x_1. Since we have assumed this does not occur, Y must fail to be G_k splitting because of the existence of some G'_k extending $p(G_k)$ and deciding G_k through $\{0,1\}^{\leq |x_1|}$, such that $p * G'_k$ does not decide $Y \upharpoonright \psi^*(Y,x_1)$ in E. Let such a G'_k be fixed.

Let q and r be extensions of $p * G'_k$ in E that strongly force incompatible values for $Y \upharpoonright \psi^*(Y,x_1)$. We claim that q is a witness to Y's being amorphous at $u_\Psi(|x_1|)$ over p. Let Y_0 be the predicate on the set of strings of length less than or equal $u_\Psi(|x_1|)$ such that q forces Y to extend Y_0. First, the condition obtained from q by changing only $G_k(x)$ from 0 to 1 or vice versa forces Y to be incompatible with Y_0 by the assumption of G_k-splitting at x. Thus, the collection of Y's values on the set of strings of length less than or equal to $u_\Psi(|x_1|)$ is not decided by any $p * q(G_i)$ with i unequal to k. It is not decided by $p * q(G_k)$, by the choice of G'_k. The above conditions are all compatible with E since E is $\mathcal{G}$-free. This verifies the claim, showing that case (3) hold, if both cases (1) and (2) fail. ◊

We define the family of strategies I(Φ, $\vec{G}$, Ψ) as follows. We will assume that I(Φ, $\vec{G}$, Ψ) is associated with a strategy I(Φ, $\vec{G}$, global) of index e. Let f be a given recursive function; let E be the extension of $E_0(s+1)$ obtained by imposing the constraint that U dominate $u_\Phi \circ u_\Psi$ and f. Given inputs $\mathcal{C} \upharpoonright s$ and $in(s+1)$, I(Φ, $\vec{G}$, Ψ) works as follows.

(1) **Transition:** Let p be any extension of $p(s)$ of the form $p(s) * U' * \vec{G}'$ in E such

that $p(U) - p(s)(U)$ is not empty and $\overrightarrow{G}'$ is compatible with $p(s)(\overrightarrow{G}) * 0$. Let x be the string of 0's of maximal length such that x is in $p(U)$. *Note, by clause (7) in* $E_0(s+1)$, *we may assume if* G_i *is named in* $p(s)$ *then* $\ell_{p(s)}(G_i) < |x|$. *By clause (5), any elements of* U' *not already in* $p(s)(U)$ *have length greater than* $gap(s)$.

(i) If there is a G_i in $\overrightarrow{G}$ such that Y is G_i-splitting in E over p, then let k be this i and go to (2).

(ii) If Y is nonsplitting at x over p in E, then go to (3).

(iii) If Y is amorphous at $u_\Psi(|x|)$ over p, then go to (4).

By Lemma 6.4, one of these conditions must hold.

(2) **Transition:** If this state is reached for the first time, let p be given from state (1). Otherwise, let p equal $p(s)$.

(i) If there is a q in E extending p with $\ell_q(U)$ less than or equal to $u_\Phi \circ u_\Psi(in(s+1))$ and x in $q(U)$ such that Y is nonsplitting at x over q, go to (3).

(ii) If Y is amorphous at $u_\Phi \circ u_\Psi(in(s+1))$ over p, go to (4).

Environment: Impose the following constraints on q. First, U dominates $u_\Phi \circ u_\Psi$. Second, if $t \leq s$ and $q(C)$ is incompatible with $p(t)(C) * 0$ then there is at most one element of $q(C) - p(t)(C)$ with second coordinate of length less than or equal to $u_\Psi(in(t+1))$.

As long as the strategy stays in this state, Y *is* G_k*-splitting at the elements of* U. *That means that the value of* G_k *at* x *can be read off from* $Y \upharpoonright \psi^*(Y,z)$ *on the elements* z *of* U *of length less than or equal to* $|x|$. *The second clause limits the number of times that the condition on* C *can be extended to be nonzero without deciding* $C \upharpoonright \psi^*(x)$*; it can only happen once.*

(3) **Environment:** Define p as in (2). Let q_0 and x be the least condition in E extending p and least element of $q_0(U)$ such that Y is nonsplitting at x over q_0 in E. Let q_1 be the least extension of q_0 in E such that for all G_i, $\ell_{q_1}(G_i)$ is less than $|x|$ and q_1 decides $Y \upharpoonright \psi(Y,x)$.

Since x *is not in the domain of any element of* $\overrightarrow{G}$, *by clause (7) of* $E_0(s+1)$, q_1 *does not strongly decide a value for any of* $A(\langle e,x\rangle)$, *any* $B(\langle e,\langle x,i,pos,neg\rangle\rangle)$, $C(\langle e,x,1\rangle)$, $C(\langle e,x,2,0\rangle)$ *or* $C(\langle e,x,2,1\rangle)$. *We will arrange that* $gap(s)$ *is larger than any other restraint on these sets coming from earlier stages, as in (2). Hence, we will be able to freely decide these values.*

Construct an extension r of q_1 with the following features.

(i) $\langle e, x, 1\rangle \in r(C)$. If y is a string other than $\langle e, x, 1\rangle$, y is not in the domain of $p(s)(C)$ and $|y|$ is less than $u_\Phi \circ u_\Psi(|x|)$ then $r(C)(y) = 0$. Let m be the value forced for $\Psi(C \oplus U \oplus Y, x)$ by $q_1 * (r(C))$ in E. *We have managed to build r forcing the value m for $\Psi(C \oplus U \oplus Y, x)$ without deciding any of the values of $G_i(x)$.*

(ii) $r(A^{(e)})(x) = 1 - m$. Further, $\langle x, 1-m, \emptyset, \emptyset\rangle$ is an element of $r(B^{(e)})$.

(iii) For each G_i mentioned in p, $r(G_i)(x) = 1 - m$.

(iv) If x is the longest element of $q_1(U)$, let x_1 be the shortest string of 0's with $|x_1| > max\{f(|x|), u_\Phi \circ u_\Psi(|x|)\}$. Let $r(U)$ extend $q_1(U)$ by setting $r(U)(y)$ equal to 0 for all strings of length less than or equal to $|x_1|$ and not in the domain of $q_1(U)$. *For the sake of $E_0(s+1)$, we include a large element of U.*

If $s+1$ is the first stage when this state is reached, then impose the constraint on the allowed conditions q, that all q must extend r and for all G_i named in q, $\{0,1\}^{\leq u_\Phi \circ u_\Psi(|x|)}$ is in the domain of q. Otherwise, no constraints are imposed.

The first time this state is reached, we impose a constraint to ensure that $A^{(e)}$ is strongly forced to be unequal to $\Psi(C \oplus U \oplus Y)$. After that, no further action is needed to satisfy the requirement.

(4) **Environment:** Let p be defined as in (2). Let q and ℓ be the least extension of p in E and least length such that q is a witness to the fact that Y is amorphous at ℓ over p. Compute a condition r extending q with the following properties.

(i) Let x_1 and x_2 be the least pair of strings of 0's of length greater than $2^{(2^{2\ell_q(U)})}$ and greater than any of $f(\ell_q(U))$, $u_\Phi \circ u_\Psi(\ell_q(U))$ and $gap(s)$ so that the extension q_1 of q obtained by making x_1 and x_2 the next two elements of U after those in $q(U)$ is in E. Set $r(U)$ equal to $q_1(U)$. *This is possible since E is U-free over the maximum of f and $u_\Phi \circ u_\Psi$.*

(ii) Set each $r(G_i)$ to be the extension of $q(G_i)$ defined on all strings of length less than or equal to $u_\Phi \circ u_\Psi(|x_1|)$, compatible with $q(G_i) * 0$. *This is compatible with clause (7) of $E_0(s+1)$, because it leaves $\ell_r(G_i) < |x_2|$.*

(iii) Set $r(C)$ to be the extension of $p(s)(C)$ of length $u_\Psi(|x_1|)$ compatible with $p(C) * 0$. This action strongly decides $C \oplus U \oplus Y \upharpoonright \psi(C \oplus U \oplus Y, x_1)$. Let m be the value forced for $\Psi(C \oplus U \oplus Y, x_1)$.

(iv) Set $r(A^{(e)})(x_1) = 1 - m$. Further, $\langle x_1, 1-m, \emptyset, \emptyset \rangle$ is an element of $r(B^{(e)})$.

This is the most dangerous move made in the definition of r. If $1 - m$ is equal to 1, then A is changed from its default value without recording that change in $C \oplus U \oplus G_i$. We will have to show that either this move is compatible with the strategies of higher priority or one of those strategies changes state. See section §7.

If $s+1$ is the first stage when this state is reached then impose the constraint on the allowed conditions q, that all q must extend r. Otherwise, no constraints are imposed.

As in (3), we strongly force $A^{(e)}$ to be different from $\Psi(C \oplus U \oplus Y)$.

6.6. LEMMA. *If C is a coherent construction that eventually respects E_0, I(Φ, $\overrightarrow{G}$, global) and I(Φ, $\overrightarrow{G}$, Ψ), then the requirement I(Φ, $\overrightarrow{G}$, Ψ) is satisfied.*

PROOF: The proof breaks into cases depending upon the limiting state of I(Φ, $\overrightarrow{G}$, Ψ). By Lemma 6.2 , $A^{(e)}$ is equal to $\Delta(B \oplus Y)$. We must show that either there is a G_i in $\overrightarrow{G}$ such that $U \oplus Y \geq_p G_i$ or $\Psi(C \oplus U \oplus Y) \neq A^{(e)}$.

Let $s_0 + 1$ be the stage during which I(Φ, $\overrightarrow{G}$, Ψ) is invoked in its initial state (1) and after which C respects I(Φ, $\overrightarrow{G}$, Ψ) and is coherent with respect to it. I(Φ, $\overrightarrow{G}$, Ψ) begins with a condition p extending $p(s_0)$ in the environment E extending $E(s_0 + 1)$. Since p is an element of $E_0(s_0+1)$, the maximal element of $p(U)$ has greater length than $\ell_p(X)$ for any set X named in p other than U. Thus, there is a string x to which we can apply Lemma 6.4. We extend the environment to E^* by imposing the constraint the U dominate $u_\Phi \circ u_\Psi$. By Lemma 6.4 , one of the three conditions (i)-(iii) suitable for transition from (1) must apply.

The first case is when I(Φ, $\overrightarrow{G}$, Ψ) reaches state (2) and stays there during every subsequent stage. To reach this state, there must be G_k in $\overrightarrow{G}$ such that Y is G_k-splitting over p at x. We claim by induction that during every stage $s + 1$ after $s_0 + 1$, $p(s)$ is G_k-splitting at every y in $p(s)(U)$ of length greater than $\ell_{p(s)}(G_k)$ and greater than $\ell_p(s_0)(U)$.

Fix s and let ℓ be the common value of $\ell_{p(s)}(G_i)$ such that $G_i \in \overrightarrow{G}$. By clause (6) in $E_0(s+1)$, the domain of $p(s)(G_k)$ is equal to $\{0,1\}^{\leq \ell}$. By Lemma 6.5, if there is a U' contained in $\{0,1\}^{\leq in(s+1)}$ and extending $p(s)(U)$ such that $p(s) * U'$ is in E^*, $x' \in U'$ and $|x'| > \ell$ such that Y is not G_k-splitting at x' over $p(s)$, then one of the two transition

conditions for leaving state (2) must hold. We are assuming this does not happen. By the coherence of $\mathcal{C}$, $p(s+1)$ is chosen by a computation of length less than or equal to $in(s+1)$ and so $\ell_{p(s+1)}(U)$ is less than $in(s+1)$. Thus, the environment in (2) inductively maintains G_k-splitting. In this case, we claim that $U \oplus Y \geq_p G_k$.

Given x, compute $G_k(x)$ from $U \oplus Y$: First, if x is not a string of 0's, then compute that $G_k(x)$ is equal to 0. Otherwise, compute $G_k(x)$ as follows.

Compute the least s such that $in(s+1) > |x|$;
Compute $p(s)$;
Compute $G_k \upharpoonright \{0^n \mid n \leq |x|\}$ by a recursion of length no greater than $|x|$.
Step 0. $G_k \upharpoonright \{0^n \mid n \leq \ell_{p(s)}(G_k)\} = p(s)(G_k)$;
Step $i+1$. Given $G_k \upharpoonright i$.
If 0^{i+1} is not an element of U,
Then return $G_k(0^{i+1}) = 0$;
Else let $C(i+1)$ be the set of conditions obtained by adding at most one point to $p(s)(C)$ having one of the forms $\langle e', x', 1\rangle$ or $\langle e', x', 2, i'\rangle$, where e' is less than or equal to $s+1$, x' is an element of U of length less than or equal to $i+1$, and i' is one of 0 or 1. For each C_0 in $C(i+1)$, compute $Y \upharpoonright \psi(C_0 \oplus U \oplus Y, 0^{i+1})$. Let Y_0 be the collection of strings queried to Y in these computations. Let $q = p(s) * (G_k \upharpoonright i * (G_k(0^{i+1}) = 0)) * \{G_j * 0 \mid j \neq k\}$.
If $Y \upharpoonright Y_0 = \Phi(q(\overrightarrow{G})) \upharpoonright Y_0$,
Then return $G_k(0^{i+1}) = 0$;
Else return $G_k(0^{i+1}) = 1$;
End if;
End if;
End step $i+1$;
End recursion;
Output the value returned for $G_k(x)$ by the recursion;
End.

By the coherence of $\mathcal{C}$, the first two steps are PTIME. For each $i+1$ less than or equal to $|x|$, the elements of $C(i+1)$ are determined by at most one of three flags for a pair $\langle e, j\rangle$ where e is less than or equal to $s+1$ and j is less than or equal to i. By the coherence

of C, $in(|x|)$ is greater than $|x|$ so $s+1$ is greater than or equal to $|x|$. At worst, the computation of $G_k(i)$ involves evaluating $\Psi(C_0 \oplus U \oplus Y, 0^i)$ for $3|x|^2+1$ many conditions C_0 on C to determine Y_0. Thus, Y_0 has at most $(3|x|^2+1)u_\Psi(|x|)$ many elements and is a subset of $\{0,1\}^{\leq u_\Psi(|x|)}$. The next step is to compute $\Phi(q(\overrightarrow{G}), z)$ for each z in Y_0. This takes no longer than $u_\Phi \circ u_\Psi(|x|)$ many steps for each z. This gives a running time on the order of $(3|x|^2+1)u_\Psi(|x|)u_\Phi \circ u_\Psi(|x|)$ for the ith step. Multiplying by $|x|$ gives the total running time. Thus, the method described above is a PTIME Turing reduction. Let Π denote this reducibility.

To see that $\Pi(U \oplus Y) = G_k$, let x be given. If x is not a string of 0's or if x is not in U then $x \notin G_k$ by the constraint imposed in E_0. The same answer is given by $\Pi(U \oplus Y, x)$. Suppose that $x \in U$. We are given that the construction respects I(Φ, $\overrightarrow{G}$, Ψ) during all stages greater than s_0 and remains in state (2). Thus, if $|x| < in(s+1)$ and $|x| > \ell_{p(s)}(G_k)$, then Y is G_k-splitting at x over $p(s) * (U \upharpoonright |x|)$.

Consider the ith step of the recursion. U's dominating $u_\Phi \circ u_\Phi$ and i in U implies that any element of C with second coordinate of length greater than i could not be queried in $\psi(C \oplus U \oplus Y, 0^i)$. During each extension of the condition on C, at most one new point can be added by clause (4) of $E_0(s+1)$. By the constraint imposed in state (2), at most one such extension could be made to C adding an element of $C \upharpoonright \psi(C \oplus U \oplus Y, 0^i)$. These one point extensions of $p(s)(C)$ were the conditions on C we checked trying to find the dependence of Y on G_k. Thus, in the course of the recursion, the set Y_0 we computed was equal to $\psi^*(Y, x)$, the set of strings that could possibly be queried as to belonging to Y in the evaluation of $\Psi(C \oplus U \oplus Y, x)$. Then, $Y \upharpoonright \psi^*(Y, x)$ and the value forced by $p(s) * (G_k \upharpoonright (|x|-1) * (G_k(x) = 0))$ for $\Phi(\overrightarrow{G}) \upharpoonright \psi^*(Y, x)$ are identical if and only if $G_k(x)$ is equal to 0. Thus, $G_k(x) = 0$ if and only if $\Pi(U \oplus Y, x) = 0$. On the other hand, if $G_k(x)$ is not equal to 0, then it must equal 1. The same holds for $\Pi(U \oplus Y, x)$, so $\Pi(U \oplus Y, x) = G_k(x)$.

The other cases are much easier to analyze from the point of view of the satisfaction of the requirement I(Φ, $\overrightarrow{G}$, Ψ). In either case, we manage to find a condition q that strongly forces $\Psi(C \oplus U \oplus Y) \neq A^{(e)}$. Forcing this inequality is sufficient to satisfy the requirement. ◊

II(Θ, Ω, i)-strategies. We design this family of strategies to work in the context of finitely many strategies of type I(Φ, $\overrightarrow{G}$, global) of higher priority. Assume that we know

the set of indices for these strategies. We also assume that a recursive function f and value for $gap(s)$ are given so that the environment imposed by the strategies of higher priority is determined by f on $\mathcal{G}$ and U

Given Θ and Ω, we build a functional Σ to satisfy

$$\Theta(A) = \Omega(B \oplus G_i) = Z \implies Z = \Sigma(C \oplus U \oplus G_i). \tag{6.7}$$

There are two possible ways to satisfy Equation 6.7. The easier of these is to find a string z and a condition q that is acceptable to the higher priority strategies such that

$$q \Vdash^* \Theta(A, z) \neq \Omega(B \oplus G_i, z). \tag{6.8}$$

Once such an inequality is established the requirement is satisfied.

The second possibility is to arrange that $\Theta(A)$ is polynomially computable from $C \oplus U \oplus G_i$. Our strategy is a more elaborate version of the exact pair strategy I(n) of section §3. We will describe a PTIME procedure so that given $s+1$, $p(s)$ and a string z with $|z|$ less than $in(s+1)$, the procedure produces a PTIME description of a predicate $A'(z)$ relative to $C \oplus U \oplus G_i$. During each stage, for each such z, we impose an environment to force

$$\Theta(A, z) = \Theta(A'(z, s+1, p(s)), z).$$

$C \oplus U \oplus G_i$ computes $\Theta(A, z)$ by first computing the relevant $s+1$ and $p(s)$ and then computing $\Theta(A'(z, s+1, p(s)), z)$.

Let E be the environment obtained by extending $E_0(s+1)$ with the constraint that U dominate f. Our strategy has three states, described as follows.

(1) **Transition:** If there are z, q, x and e such that $|z| < in(s+1)$, $q <_E p(s)$ with $\ell_q(U) < in(s+1)$, $x \in q(U)$ with $|x| \geq gap(s)$, either e is less than $s+1$ and not the index of a strategy I(Φ, $\overrightarrow{G}$, global) of higher priority or e is such an index and $\Phi(\overrightarrow{G})$ is amorphous at $\log(\log(|x|))/2$ and

$$\Theta(p(s) * (A * 0), z) \neq \Theta(p(s) * [(A(\langle e, x \rangle) = 1) * 0], z)$$

then,

(i) if e is the index of a higher priority strategy, go to (2);

(ii) if e is not the index of a higher priority strategy, go to (3).

Environment: Recall our notation, u_Θ is a polynomial dominating the running time of Θ. We impose the following constraints on the extensions q of $p(s)$. First, for all t less than s, if there are e and x such that $p(t+1)(A) - p(t)(A)$ is $\{\langle e, x\rangle\}$ and there is no flag in $p(t+1)(C)$ with first coordinate equal to $\langle e, x\rangle$, then impose these conditions: $q(A)$ is equal to $p(t+1)(A) * 0$ and $q(C)$ is equal to $p(t)(C) * 0$ on all strings of length less than or equal to $u_\Theta(in(t+1))$.

Second, one of the following conditions holds.

(1.i) There is a z in $q(U)$ and an e less than or equal to $s+1$ such that $\langle e, z, 1\rangle$ is an element of $q(C) - p(s)(C)$. Then, there can be no element of $q(A) - p(s)(A)$ other than $\langle e, z\rangle$. For all i less than or equal to $s+1$, whether $\langle e, z\rangle$ belongs to $q(A)$ is recorded in $q(G_i)$ by

$$\langle e, z\rangle \in q(A) \iff z \in q(G_i).$$

Lastly, q must decide A for all strings with length less than or equal to $u_\Theta(in(s+1))$ and decide U and G_i on all strings of length less than or equal to $|z|$. *Given a string z of length less than or equal to $in(s+1)$, $C \oplus U \oplus G_i$ computes $\Theta(A'(z, s+1, p(s)), z)$ to simulate $\Theta(A, z)$. Given a query to A about $\langle e, z\rangle$, $C \oplus U \oplus G_i$ can test to see if case (1.i) was invoked at $\langle e, z\rangle$ by examination of C. If so, then $C \oplus U \oplus G_i$ can directly compute whether $\langle e, z\rangle$ belongs to A and return this answer.*

(1.ii) There is a z in $q(U)$, an e less than or equal to $s+1$ and j either 0 or 1 such that $\langle e, z, 2, j\rangle \in q(C) - p(s)(C)$. There can be no element of $q(A) - p(s)(A)$ other than $\langle e, z\rangle$. Further,

$$\langle e, z\rangle \in q(A) \iff j = 1.$$

Lastly, q must decide A on every string of length less than or equal to $u_\Theta(in(s+1))$ and decide U and G_i on every string of length less than or equal to $|z|$. *Case (1.ii) is similar to (1.i) except that the value of A at $\langle e, z\rangle$ is directly coded into C.*

(1.iii) Either $q(A)$ is compatible with $p(s)(A) * 0$ or for all z with $|z|$ less than or

equal to $in(s+1)$,

$$\Theta(q(A)*0,z) = \Theta(p(s)(A)*0,z).$$

E_0 imposes the constraint that if (1.i) and (1.ii) do not hold then $q(C)$ is compatible with $p(s)(C)*0$, i.e. no flags are set in C. For all z with $|z| \leq in(s+1)$, the value of $\Theta(A,z)$ can be found by computing $\Theta(p(s)(A)*0,z)$. $p(s)(A)*0$ is the value for $A'(z,s+1,p(s))$ in the absence of C-flags.

(2) **Environment:** If this is not the first stage when this stage was reached, then no constraints are imposed.

Otherwise, let z_0, q_0, x_0 and e_0 be fixed to be the least quadruple satisfying the transition condition. Suppose that e_0 is the index for $\mathrm{I}(\Phi_0, \overrightarrow{G}_0, \text{global})$. Construct an extension r of q_0 as follows.

Let ℓ equal $\log(\log(|x_0|))/2$. Let q_1 extend q_0 and amorphously decide $Y \upharpoonright \{0,1\}^{\leq \ell}$. Let Y_0 be the subset of $\{0,1\}^{\leq \ell}$ such that q_1 forces $Y \upharpoonright \{0,1\}^{\leq \ell} = Y_0$. Let $\overline{Y_0}$ be $\{0,1\}^{\leq \ell} - Y_0$. We can assume that q_1 and $p(s)$ agree on A, B and C, since the definition of being amorphous does not depend on these arguments.

(i) Let $r(G_i)$ equal $q_1(G_i)$.

(ii) Extend $q_1(B)$ to $r(B)$ by setting $r(B^{(e)})(\langle x_0, 1, Y_0, \overline{Y_0}\rangle)$ equal to 1 and making $r(B^{(e)})$ equal 0, for every other string of length small enough to be queried in the computation of $\Omega(B \oplus G_i, z_0)$. This forces a value m for $\Omega(B \oplus G_i, z_0)$.

(iii) Let A_0 be whichever of $p(s)(A)*0 \upharpoonright \{0,1\}^{\leq u_\Theta(in(s+1))}$ or $p(A)*(A(\langle e_0, x_0\rangle) = 1)*0 \upharpoonright \{0,1\}^{\leq u_\Theta(in(s+1))}$ that forces $\Theta(A, z_0) \neq m$. Since the two conditions force incompatible values for $\Theta(A, z_0)$, they cannot both force the value to be m. Let $r(A)$ equal A_0.

We have ensured that r forces $\Theta(A, z_0) \neq \Omega(B \oplus G_i, z_0)$.

(iv) Since $\Phi(\overrightarrow{G})$ is amorphous at ℓ over q_0, we can find a condition $\overrightarrow{G}_0$ on the set of G_i named in $p(s)$, compatible with $q_0 * q_1(U)$, such that $\overrightarrow{G}_0(G_i) = r(G_i) = q_1(G_i)$ and $\overrightarrow{G}_0$ forces that $\Phi(\overrightarrow{G})$ is incompatible with Y_0. For j not equal to i, let $r(G_j)$ be defined by

$$r(G_j) = \begin{cases} q(G_j)*0, & \text{if } r(A)(x_0) = 1; \\ \overrightarrow{G}_0(G_j), & \text{otherwise.} \end{cases}$$

Thus, we force $A(x_0)$ is equal to 1 if and only if Y is compatible with Y_0. By putting $\langle x_0, 1, Y_0, \overline{Y_0}\rangle$ into $B^{(e)}$, we force that Y is compatible with Y_0 if and only if $\Delta(B \oplus Y, x_0)$ is equal to 1. Thus, we satisfy the constraints of the type I global strategy of index e_0. At other indices, A and $\Delta(B \oplus Y)$ were given their default value 0. This satisfies the constraints imposed by the I(Φ, $\vec{G}$, global) strategies.

Impose the constraint that the constraint on the allowed conditions q, that all q must extend r.

(3) **Environment:** If this is not the first stage when this stage was reached then no constraints are imposed.

Otherwise, let z_0, q_0, x_0 and e_0 be the least quadruple satisfying the transition condition. Construct an extension r of q_0 as in (2), without the steps concerned with respecting a type I global strategy of index e_0.

(i) For all G_i named in q_0, let $r(G_i)$ equal $q_0(G_i) * 0$ on all strings of length less than or equal to $u_\Theta(in(s+1))$.

(ii) Let $r(B)$ equal $q_0(B) * 0$. This forces a value m for $\Omega(B \oplus G_i, z_0)$.

(iii) Let A_0 be whichever of $p(s)(A)*0 \upharpoonright \{0,1\}^{\leq u_\Theta(in(s+1))}$ and $p(A)*(A(\langle e_0, x_0\rangle) = 1) * 0 \upharpoonright \{0,1\}^{\leq u_\Theta(in(s+1))}$ gives the inequality $\Theta(A, z_0) \neq m$. Let $r(A)$ equal A_0.

(iv) Let $r(C)$ extend $q_0(C)$ by being equal to 1 at $\langle e, x_0, 2, A_0(\langle e, x\rangle)\rangle$ and being 0 for every other string corresponding to a flag for $\langle e', x'\rangle$ where e' and x' have length less than $u_\Theta(in(s+1))$.

Impose the constraint that the constraint on the allowed conditions q, that all q must extend r. *Again, we force $\Theta(A, z_0) \neq \Omega(B \oplus G_i, z_0)$.*

6.9. LEMMA. *If $\mathcal{C}$ is a coherent construction that eventually respects II(Θ, Ω, i) then requirement II(Θ, Ω, i) is satisfied by the sets produced.*

PROOF: If $\Theta(A)$ is not equal to $\Omega(B \oplus G_i)$, then the requirement is satisfied. Assume that these two are equal. We will show that their common value is in PTIME($C \oplus U \oplus G_i$). Let s_0 be the stage after which $\mathcal{C}$ is coherent for and respects II(Θ, Ω, i). Work above s_0. Since $\Theta(A)$ is equal to $\Omega(B \oplus G_i)$, the strategy stays in state (1) during every stage after s_0.

Consider the function defined relative to $C \oplus U \oplus G_i$ from a table of values for $\Theta(A)$

on arguments of length less than or equal to $in(s_0)$ as follows.

If $|x| \leq in(s_0)$;

Then look up $\Theta(A, x) = answer$ in table;

Else:

Compute the least s such that $in(s+1) \geq |x|$;

Compute $p(s)$;

Simulate the computation of $\Theta(A, x)$ to obtain *answer* responding to a query x as follows:

Cases:

If x is in the domain of $p(s)(A)$, then answer $p(s)(A)(x)$;

If $x \neq \langle e, z \rangle$ with e coding an integer less than or equal to $s+1$ and z in U, answer 0;

Else, for query $\langle e, z \rangle$:

Cases:

If $\langle e, z, 1 \rangle \in C$, answer $G_i(z)$;

If $\langle e, z, 2, i \rangle \in C$, answer i;

Else, answer 0;

End cases;

End cases;

Output *answer*;

End.

By the coherence of $\mathcal{C}$, the first two steps are uniformly PTIME. The simulation of $\Theta(A, x)$ is PTIME($C \oplus U \oplus G_i$) since there is a bound on the number of steps required to respond to a query. The only queries are to C, U and G_i.

Let $s+1$ be least such that $in(s+1)$ is greater than $|x|$. Suppose that the simulation does not reproduce the computation of $\Theta(A, x)$. This must occur because of a query that does not receive the correct response. If the query is in the domain of $p(s)(A)$, not of the correct form $\langle e, z \rangle$, or is flagged in C then the response will be correct (proof by induction on t, using $p(t) \in E_0(t)$ and the environment produced in state (1)). The only remaining case is when $\langle e, z \rangle$ is not flagged in C yet $\langle e, z \rangle \in A$. Now, let s_1 be the least stage such that $p(s_1)(A)$ is not compatible with $p(s)(A) * 0$. If either of cases (1.i) or (1.ii) held during stage s_1, $\langle e, z \rangle$'s being in A requires a flag's being in C.

Thus, case (1.iii) must have been in effect during every stage between $s+1$ and s_1. By the constraints enforced in (1.iii), every $p(t)(C)$ is compatible with $p(s)(C)*0$; so the only points appearing in $\vartheta(A,x)$ that are flagged by C are those that are flagged by $p(s)(C)$ giving correct values about $A(s)$. Thus, the simulation responds to every query either by reference to $p(s)(A)$ or with answer 0. Since case (1.iii) held between stage $s+1$ and s_1, $p(s_1-1)(A)$ and $p(s_1-1)(C)$ are compatible with $p(s)(A)*0$ and $p(s)(C)*0$. Since, $p(s_1)(A)$ is not compatible with $p(s)(A)*0$, it is also not compatible with $p(s_1-1)(A)*0$. By the conditions of 1.iii, $p(s_1)$ was required to satisfy

$$\begin{aligned}\Theta(p(s_1)(A)*0,x) &= \Theta(p(s_1-1)(A)*0,x)\\ &= \Theta(p(s)(A)*0,x).\end{aligned}$$

By the constraint imposed in (1) during later stages t, $p(t)(A)$ is compatible with $p(s_1)(A)*0$ on all elements of $\{0,1\}^{\leq u_\Theta(in(s_1))}$. This includes $A \upharpoonright \vartheta(A,x)$. So, the true value is equal to the one obtained by the simulation. $\Diamond$

The environment created by a strategy of type II is very restrictive. It limits our ability to put $\langle e,x\rangle$ into $A^{(e)}$ to one of two cases: either, we flag $\langle e,x\rangle$ in C and record whether x is in $A^{(e)}$ in $C\oplus U\oplus G_i$ or, for all z such that $|z|$ is too small to compute $A(\langle e,x\rangle)$ directly, $\Theta(A,z)$ does not depend on whether x is in $A^{(e)}$

III(Θ, j)-strategies. We must satisfy

$$\Theta(\oplus_{i\neq j}G_i\oplus U)\neq G_j.$$

The strategy to satisfy this requirement is routine. Assume, as earlier, that this strategy is working in an environment E extending $E_0(s+1)$ that is determined by f on $\mathcal{G}$ and U. Its only effect on the environment is to chose an extension q of $p(s)$ compatible with E such that q strongly forces $\Theta(\oplus_{i\neq j}G_i\oplus U)\neq G_j$ based on the assumption that any G_k not named in q is empty. III(Θ, j) requires that $p(s+1)$ be an extension of q.

In E, put such a q together as follows. First, extend U to add two new consecutive elements x and x_1 where x_1 is too large to be queried in the evaluation of $\Theta(\oplus_{i\neq j}G_i\oplus U,x)$ and $|x|$ is larger than $gap(s)$; then, decide all the G_i except for G_j on $\{0,1\}^{\leq|x|}$ thereby forcing a value for Θ; finally, choose $G_j(x)$ different from the value forced.

6.10. LEMMA. *If $\mathcal{C}$ is a coherent construction that eventually respects III(Θ, j), then requirement III(Θ, j) is satisfied by $\mathcal{C}$.*

PROOF: First, we are assuming that the construction goes through every stage. In particular, when the strategy acts, the diagonalizing condition q exists or the construction would terminate. In our construction, this is ensured by the compatibility of our strategies; see Section §7.

The only point to worry about is that some of the G_i referred to in $\theta(\oplus_{i\neq j}G_i \oplus U)$ may not be explicitly named in the environment. But by the condition imposed in $E_0(s+1)$ clause (6), if G_i is not named in $p(s)$ then any string in G_i must have length greater than or equal to $\ell_{p(s)}(U)$. Thus, the computation applies to $\oplus_{i\neq j}G_i \oplus U$. $\Diamond$

IV(i, j, coding)-strategies. For each j and i with $j \leq_{P_N} i$, we build a Λ so that

$$\Lambda(L \oplus G_i) = G_j. \tag{6.11}$$

The strategy is similar to the global type II strategies. Namely, we will define the reduction Λ and restrict the environment to those conditions that do not strongly force the negation of Equation 6.11. Let e be the index of this requirement.

Environment: The only allowed conditions at stage $s+1$ are those q such that, for all x not in the domain of $p(s)(G_j)$ and x in the domain of $q(G_j)$

$$q \Vdash^* G_j(x) = \begin{cases} L(\langle e, x\rangle), & \text{if } L(\langle e, x, 1\rangle) = 0 \\ G_i(x), & \text{otherwise.} \end{cases} \tag{6.12}$$

We further require that x is in the domain of $q(G_j)$ if and only if $\langle e, x\rangle$ and $\langle e, x, 1\rangle$ are in the domain of $q(L)$ and x is in the domain of $q(G_i)$.

We define $\Lambda(L \oplus G_i, x)$ to equal the right hand quantity in Formula 6.12. The fact that this strategy is sufficient to satisfy its associated requirement is easily verified.

IV(Θ, i, j)-strategies. This is another case where the strategies use direct diagonalization. Here, we work in an environment extending $E_0(s+1)$, with finitely many type IV coding constraints that is determined by f on $\mathcal{G}$ and U. For $j \not\leq_{P_N} i$, the requirement is

$$\Theta(G_i \oplus L \oplus U) \neq G_j. \tag{6.13}$$

As in III(Θ, j), we impose the one time only constraint that $p(s+1)$ be chosen to extend a condition q that strongly forces an instance of Equation 6.13.

We build q as follows. First, chose x and x_1 as in III(Θ, j); decide all the elements G_k of $\mathcal{G}$ named in $p(s)$ such that $k \leq_{P_N} i$ all strings of length less than or equal to $u_\Theta(|x|)$; for each $k \leq_{P_N} i$ subject to a coding constraint of type IV with index e' and each x' added to the domain of G_k, extend L so that $L(\langle e', x', 1\rangle)$ is equal 0 and $L(\langle e', x'\rangle$ is equal to $G_k(x')$; next, for all other indices e' for type IV coding strategies, set $L(\langle e', x', 1\rangle)$ equal to 1 for all strings x' of length less than or equal to $u_\Theta(|x|)$. These steps decide $\Theta(G_i \oplus L \oplus U, x)$.

Decide $G_j(x)$ to be different from the value forced for $\Theta(x)$ and to be defined on all strings of length less than or equal to $|x|$. For all G_k named in $p(s)$ but not decided above, extend $p(s)(G_k)$ to be equal to G_j on all points added to the domain. Thus, we ensure that $p(s+1)$ satisfies Equation 6.13 while respecting the higher priority coding strategies.

The proof that these strategies are sufficient to ensure that their associated requirements are satisfied is straight forward.

V-strategies. Suppose that m is an integer in the sense of P_N. Let m' be the integer that codes m's successor in P_N. We will describe the type V strategies with the assumption that m is even. The definitions for the case when m is an odd integer are exactly analogous, replacing 0 by 1 below.

V(m, coding)-strategies. We must build Γ and Δ so that

$$\Gamma(P_0 \oplus G_m) = \Delta(Q_0) = G_{m'}.$$

Environment: The functional Γ is defined relative to $P_0 \oplus G_m$ exactly like Λ was defined relative to $L \oplus G_i$ in IV(i, j, coding). Namely, we extend the environment to only allow conditions q such that

$$q \Vdash^* G_{m'}(x) = \begin{cases} P_0(\langle e, x\rangle), & \text{if } P_0(\langle e, x, 1\rangle) = 0 \\ G_m(x), & \text{otherwise.} \end{cases} \tag{6.14}$$

We further require that x is in the domain of $q(G_{m'})$ if and only if $\langle e, x\rangle$ and $\langle e, x, 1\rangle$ are in the domain of $q(P_0)$ and x is in the domain of $q(G_m)$.

Δ is defined by eventually requiring that $Q_0^{(m')}$ be equal to $G_{m'}$ at all remaining strings. That is, restrict the environment to the conditions that do not strongly force an inequality between these two.

Again, we omit the proof that these strategies ensure the satisfaction of their requirements.

V(m, Φ, Ψ)-strategies. This strategy must arrange

$$\Phi(G_m \oplus P_0 \oplus U) = \Psi(Q_0 \oplus U) = Z \implies \Delta(G_{m'} \oplus U) = Z.$$

This strategy is another elaboration on the type I strategies of section §3. Either, we strongly force an inequality between $\Phi(G_m \oplus P_0 \oplus U)$ and $\Psi(Q_0 \oplus U)$, or for each s, their common value at any x with $|x| \leq in(s+1)$ depends only on $p(s)$ and $G_{m'} \oplus U$. Let f be a recursive function and let E^* be the extension of $E_0(s+1)$ requiring that U dominate f and imposing the constraint associated with V(m, coding) with index e.

(1) **Transition:** If there is an x with $|x| < in(s+1)$ and a condition q in E^* extending $p(s)$ such that, $q(P_0)$ is compatible with $p(s)(P_0) * 1$ for every string of the form $\langle e', x, 1\rangle$ where e' is not equal to e; and

(6.15) $$q \Vdash^* \Phi(G_m \oplus P_0 \oplus U, x) \neq \Psi(Q_0 \oplus U, x)$$

then, go to (2).

Environment: No constraints.

(2) **Environment:** If this is the first stage when this state is reached, require that $p(s+1)$ extend the least q as in Formula 6.15. Otherwise, no constraints.

6.16. LEMMA. *Suppose that C is a coherent construction that eventually respects V(m, Φ, Ψ). Then, requirement V(m, Φ, Ψ) is satisfied by the sets produced.*

PROOF: Suppose that $\Phi(G_m \oplus P_0 \oplus U)$ is equal to $\Psi(Q_0 \oplus U)$. Consider the following Turing reduction. Let $p(s)(Q_0) * G_{m'} * 0$ denote the predicate extending $p(s)(Q_0)$ obtained by letting the m'th column be equal to $G_{m'}$ and the rest of the predicate be empty. Let s_0 be the stage such that V(m, Φ, Ψ) is respected by the construction after stage s_0. Given input x, and a table of values for $\Psi(Q_0 \oplus U)$ at arguments of length less than $in(s_0+1)$ compute $\Delta(G_{m'} \oplus U, x)$ as follows.

If $|x| \leq in(s_0)$,

Then let *answer* equal the value of $\Psi(Q_0 \oplus U, x)$ indicated in the table;
Else compute as follows:
Compute the least s such that $in(s+1) \geq |x|$;
Compute $p(s)$;
Compute $\Psi(p(s)(Q_0) * G'_m * 0 \oplus U, x) = answer$
End else;
End if;
Output *answer*;
End.

Since Ψ is in PTIME and $\mathcal{C}$ is coherent, Δ is also PTIME. If there is an x such that $\Delta(G_{m'} \oplus U, x) \neq \Psi(Q_0 \oplus U, x)$ then there is a stage after s_0 such that there are two conditions on Q_0 forcing incompatible values for $\Psi(Q_0 \oplus U, x)$. These are q_0, the one found during the execution of Δ and q_1, the one that applies to Q_0. These two conditions are only different on the coordinates of Q_0 other than the m'th coordinate. They decide values for sets G_k, only when k represents an odd integer in P_N.

We consider the predicate $p(s)(P_0) * 1$. This predicate gives the responsibility of satisfying Formula 6.14 to the G_k where k is an even integer in P_N. Let G^*_m be the predicate extending $p(s)(G_m)$ that is equal to $G_{m'}$ at every string y such that $P(\langle m, y, 1\rangle) = 1$. Let r be the minimal extension of $p(s)$ consistent with setting $r(P_0)$ equal to $p(s)(P_0) * 1$ and $r(G_m)$ equal to G^*_m that strongly forces a value for $\Phi(G_m \oplus P_0 \oplus U)$. Let k be equal to the value forced by r. One of the two conditions q_i forces $\Psi(Q_0 \oplus U, x)$ to be unequal to k. Thus, the transition condition is met and an inequality must have been established between $\Phi(G_m \oplus P_0 \oplus U)$ and $\Psi(Q_0)$. ◇

§7. Compatibility of the Strategies
Proof of Theorem 4.9

Compatibility between strategies. In the previous section, we analyzed the strategies in isolation. The next step is to show that the strategies can be combined in a priority construction. We begin by showing that each family of strategies is dense in any environment imposed by a sequence of outcomes of strategies of higher priority.

The simplest type of environment requires only that the construction extend a condition chosen to strongly force some existential statement. For example, such an environment is

created by the one time only diagonalization strategies: III(Θ, i, j), making $\Theta(\oplus_{i\neq j}G_i \oplus U) \neq G_j$, and IV($\Theta$, i, j), making $\Theta(G_i \oplus L \oplus U) \neq G_j$. Similarly, a strategy that is waiting for an appropriate situation, produces a single condition environment, if its transition condition is ever satisfied. This is the case for I(Φ, $\vec{G}$, Ψ), if it reaches either state (3) or (4); II(Θ, Ω, i), if it reaches state (2) or (3); and for V(m, Φ, Ψ), if it reaches state (2). We will refer to these collectively as the *finite outcomes.*

Each strategy S is designed to work under the hypothesis that every strategy of higher priority that has a finite outcome reaches its limit state during a stage before S is initially activated. The remaining strategies of higher priority than S shape the environment within which S is constrained to work. They have a common feature: by one means or another, they ensure that one set computes another one. Of course, no single condition can force such an equality, so a global change in the environment is necessary. The global outcomes are divided into two cases.

The first type of global outcome is called a *definite global outcome.* Here, some coding constraint is imposed with absolute control. The definite global outcomes are given by the coding strategies in their unique state: I(Φ, $\vec{G}$, global), IV(i, j, coding), and V(m, coding). These strategies build PTIME functionals and ensure that certain equalities hold for every string, by controlling both sides of the equations.

The second type of global outcome is called a *conditional global outcome.* In this type of outcome, the strategy ensures that some equation will hold for all strings, provided that the strategy's transition condition is never realized. These outcomes are I(Φ, $\vec{G}$, Ψ), staying in state (2); II(Θ, Ω, i), staying in state (1); and V(m, Φ, Ψ), staying in state (1).

The next lemma gives a summary of the environment produced when no strategy changes state.

7.1. LEMMA. *Suppose that $C\restriction s$ is given. Let $\vec{S}$ and $\vec{\sigma}$ be length n sequences of strategies beginning with E_0 and continuing with types I to V, and associated states corresponding to global outcomes, respectively, appropriate to extend $C\restriction s$. Let $\vec{m}$ be the input sequence determined from these parameters and an input $OUT(s)$. Assume none of the elements of $\vec{S}$ change state with this input and let E be the defined environment. Whether q belongs to E is uniformly characterized in terms of $C\restriction s$, $\vec{S}$, $\vec{\sigma}$, and* OUT *as follows.*

(1) *E is a subset of $E_0(s+1)$.*

(2) *E is contained in the intersection of the environments imposed by strategies with definite global outcomes in* $\vec{\sigma}$.

(3) *For each I(*Φ, $\overrightarrow{G}$,Ψ*) in* $\overrightarrow{S}$ *in state (1) with index e, if* $t \leq s$ *and* $q(C)$ *is incompatible with* $p(t)(C) * 0$ *then there is at most one element of* $q(C) - p(t)(C)$ *with second coordinate of length less than or equal to* $u_\Psi(in_e(t))$.

(4) *For each II(*Θ, Ω, *i) in* $\overrightarrow{S}$ *in state (1) with index e,* $q(A)$ *and* $q(C)$ *respect clauses (1.i)-(1.iii) of that state with regard to the compatibility between the values of* $q(A)$ *and the existence of flags in* $q(C)$.

(5) *Further, if* $t \leq s$, *if there are e and x such that* $p(t+1)(A) - p(t)(A)$ *is* $\{\langle e, x\rangle\}$ *and there is no flag in* $p(t+1)(C)$ *with first coordinate equal to* $\langle e, x\rangle$, *then:* $q(A)$ *and* $q(C)$ *are compatible with* $p(t+1)(A) * 0$ *and* $p(t)(C) * 0$, *respectively, on* $\{0,1\}^{\leq u_\Theta(in_e(t+1))}$.

(6) *There is a recursive function f, depending only on* $\vec{\sigma}$, *such that E imposes the constraint that U dominate f.*

PROOF: The proof is by examination of all of the cases. The initial environment is covered by (1). The definite global outcomes are covered in (2). It remains to show that (3), (4), (5) and (6) summarize the possible effect of finitely many conditional global outcomes.

(4) and (5) cover the effects of conditional global outcomes from type II strategies. The remaining outcomes come from type I and V strategies. The additional effect of a type I conditional outcome is to impose the condition given in (3) and to require that U dominate some additional recursive function, uniformly obtained from an index for the strategy. The latter effect is included in (6). The environment created by a type V strategy in its conditional global outcome (state (1)) is the same as its input environment. By induction, the intersection of the imposed environments is characterized as above. ◇

7.2. COROLLARY. *Use the same notation as in Lemma 7.1. First, E is determined by f on* $\mathcal{G}$ *and U. Further, for all m greater than gap(s), any* $u_\Psi(in_e(t))$ *referred to in (2) and any* $u_\Theta(in_e(t))$ *referred to in (3),* $p(s) * (U(0^m) = 1)$ *does not decide any values of* $A(\langle e, 0^m\rangle)$, $B(\langle e, \langle 0^m, i, pos, neg\rangle\rangle)$, $C(\langle e, 0^m, 1\rangle)$ *or* $C(\langle e, 0^m, 2, i\rangle)$, *not already decided by* $p(s) * (U(0^m) = 1)$ *in the environment determined by* $E_0(s+1)$, *the definite global strategies and the clauses (1.i) to (1.iii) in the type II(*Θ, Ω, *i) strategies in* $\overrightarrow{S}$.

PROOF: By Lemma 7.1, the only constraints on the U coordinate of a condition in E come from clause (1), saying that $q(U) \subset \{0\}^*$ and clause (4), saying that U dominate f. For the second claim, the only constraints in E on A, B or C, other than the ones mentioned above, are summarized in clauses (3) and (5) of Lemma 7.1. These are limited to strings of length less than m. $\Diamond$

7.3. LEMMA. *Retain the notation as in Lemma 7.1. Assume $gap(s)$ is the supremum of $u_\Psi(in_e(t))$ and $u_\Theta(in_e(t))$ for $t \leq s$ where Ψ and Θ come from type I and II elements of $\overrightarrow{S}$, respectively. There is a recursive method, which when applied to the inputs which produces a condition r in* out *many steps. Further, if $\overrightarrow{S}$ is executed using the input sequence defined from $\mathcal{C} \restriction s$, $\overrightarrow{S}$, $\vec{\sigma}$ and* OUT $= OUT(s) +$ out, *then one of*

(1) *an element S_i of $\overrightarrow{S}$ changes state;*

(2) *r is an element of the defined environment and for all X named in r, $\{0,1\}^{\leq \ell}$ is contained in the domain of $r(X)$.*

PROOF: We specify the values of $r(X)$ over $p(s)(X)$; we leave it to the reader to complete the definition of r by setting values for ℓ_r.

By Lemma 7.1 and Corollary 7.2, there is a recursive f, depending only on $\overrightarrow{S}$ and $\vec{\sigma}$, such that E is determined by f on $\mathcal{G}$ and U. First, extend $p(s)(U)$ by, first, adding an element 0^m with m greater than $f(\ell_{p(s)}(U))$, $gap(s)$ and ℓ; second, setting $r(U)(x)$ equal to 0 at every string not in the domain of $p(s)(U)$ and of length less than or equal to m. We will only define other predicates on strings of length less than m. This action satisfies $E_0(s+1)$.

Let $r(G_i)$ extend $p(s)(G_i)$ by setting $r(G_i)(z)$ equal 0, for all G_i named in p and all strings z of length less than or equal to ℓ not in the domain of $p(s)(G_i)$.

Let $r(Q_0)$ and $r(Q_1)$ be defined so as to maintain the two equalities:

$$Q_0 = \oplus\{G_i \mid i \text{ is an odd integer in } P_N.\}$$
$$Q_1 = \oplus\{G_i \mid i \text{ is an even integer in } P_N.\}.$$

If m represents an odd integer in P_N, $|\langle m,x\rangle|$ is less than or equal to ℓ and G_m is not named in p, set $r(Q_0)(\langle m,x\rangle)$ equal to 0. Similarly, if m represents an even integer in P_N, $|\langle m,x\rangle|$ is less than or equal to ℓ and G_m is not named in p, set $r(Q_1)(\langle m,x\rangle)$ equal to 0.

Define $r(L)$, $r(P_0)$ and $r(P_1)$ so that all values to be coded are correctly recorded in L, P_0 and P_1. That is, suppose that e is an index for a coding strategy IV(i, j, coding) or V(m, coding) in $\vec{S}$. Define the appropriate predicate (L, P_0 or P_1) to equal 0 at each $\langle e, x, 1\rangle$ where r has added x to the domain of the set X to be coded (G_j or $G_{(2m)'}$, respectively) and to have value $X(x)$ at $\langle e, x\rangle$. For strings z, not of the form $\langle e, x\rangle$ and of length less than or equal to some element of the domain of r at a coding predicate (L, P_0 or P_1), set the coding predicate equal to 0 at z. Thus, r fulfills all of the definite global constraints.

For a conditional global constraint coming from a strategy II(Θ, Ω, i) in state (1), the imposed constraint can be respected by clause (1.iii): A, B and C are given their default value 0 for all strings of length less than ℓ where they were not already defined by $p(s)$.

By construction, r is an extension of $p(s)$ such that if X is named in r then $\ell_r(X)$ is greater than or equal to ℓ. We ensured that r lie in E by explicitly respecting all of the constraints imposed by $\vec{S}$ in states $\vec{\sigma}$ given input $\vec{in}$. ◊

7.4. DEFINITION. Let $\vec{\mathcal{S}}$ a recursive list beginning with E_0 followed by the families of strategies of type I to V such that each family I(Φ, $\vec{G}$, Ψ) is preceded by I(Φ, $\vec{G}$, global). Further, for all n, $\mathcal{S}_n$ names only sets from the set of basic parameters or from $\{G_i \mid i \leq n\}$.

To check a fine point, since the set of integers and the arithmetic operations coded in P_N are recursive, there is a recursive method to see which of the type IV strategies should be used for the pair i and j, and whether the type V strategy with indices m and m' should be used.

Having fixed the priority ordering, we will now define $gap(s)$ in terms of the stage $s+1$ inputs.

7.5. DEFINITION. Given the first s stages of a construction using strategies from $\vec{\mathcal{S}}$, let $gap(s)$ be the supremum of $u_\Psi(in(t))$ and $u_\Theta(in(t))$ for $t \leq s$, where Ψ and Θ come from type I and II strategies in $\vec{\mathcal{S}} \restriction s+1$.

7.6. PROPOSITION. *$\vec{\mathcal{S}}$ is compatible.*

PROOF: Let $\vec{S}$ and $\vec{\sigma}$ be length n sequences of strategies and associated states $\vec{\sigma}$ with $S_i \in \mathcal{S}_i$. By Lemma 7.1, unless some element of $\vec{S}$ changes state, the environment imposed by $\vec{S}$ in states $\vec{\sigma}$ is characterized in terms of the indices of the strategies and

U's dominating a recursive function that is uniformly obtained from $\overrightarrow{S}$ and $\vec{\sigma}$ and a bounded constraint on A and C.

By examining the descriptions of the families of strategies of type I to V in section §6, every strategy was recursively described from these terms, without mention of the bounded constraint. Thus, there is a recursive and canonical method to extend $\overrightarrow{S}$ and $\vec{\sigma}$ by an element in $\mathcal{S}_{n+1}$ in state (1). It remains to show that the resulting sequence is compatible.

To show compatibility, we must give a uniformly recursive method to produce a condition r, given the state of a construction $\mathcal{C}\restriction s$ that has respected $\overrightarrow{S}$. Letting *out* be the number of steps needed to compute r, we must further show that either r is an element of the environment recursively defined from $\mathcal{C}\restriction s$ and $OUT(s)+out$ or one of the elements $\overrightarrow{S}*S$ changes state from $\vec{\sigma}*(1)$ with this input.

Since $\overrightarrow{S}$ was respected by $\mathcal{C}\restriction s$ during stage s, we may assume that any finite outcome due to an element of $\overrightarrow{S}$ during stage $s+1$ is associated with a change in state. Lemma 7.3 gives exactly the method we need for the case when every state visited during stage $s+1$ is associated with a global outcome. By induction, we need only show that no finite outcome of S requires that $p(s+1)$ extend a condition that is not in the environment subsequently computed during stage $s+1$, without some element of $\overrightarrow{S}$ changing state. That is, no finite outcome of S is incompatible with the environment subsequently imposed by $\overrightarrow{S}$ in state $\vec{\sigma}$. We examine the cases in increasing order.

To fix some notation, let E be the environment extending $E_0(s+1)$ calculated in Lemma 7.1 under the assumption that no element of $\overrightarrow{S}$ changes state or has a finite outcome. Let f be the function associated with $\overrightarrow{S}$ and $\vec{\sigma}$ such that E is determined by f on $\mathcal{G}$ and U.

The first case is when S is strategy of type I(Φ, $\overrightarrow{G}$, Ψ). As usual let Y denote $\Phi(\overrightarrow{G})$. There are two possible finite outcomes for a strategy of this type, it could reach either state (3) or (4). Let p be the condition used upon entering the finite outcome state. Recall, p is either $p(s)$ or an extension of $p(s)$ obtained by adding a new element to U that is longer than $gap(s)$ respecting the domination of f by U. Let Y denote $\Phi(\overrightarrow{G})$ for the discussion of this case.

In the nonsplitting state (3), I(Φ, $\overrightarrow{G}$, Ψ) uses a condition q in E and a string x such that q decides U and Y on all strings that could be queried during the computation of

$\Psi(C \oplus U \oplus Y, x)$ but has $\ell_q(G_i) < |x|$ for all $G_i \in \overrightarrow{G}$. If this state was reached during the initialization of I(Φ, $\overrightarrow{G}$, Ψ), it is explicitly ensured that $|x| > gap(s)$. If this state is reached after initialization, we cannot have x in $p(s)(U)$. If x were in $p(s)(U)$, the transition condition for I(Φ, $\overrightarrow{G}$, Ψ) would have been satisfied during the previous stage; by the assumption that $C \restriction s$ respects S, S would have changed state during stage s. Thus, x must be a new element of U. In the transition condition, we explicitly ensured that $|x|$ would be greater than $gap(s)$.

An extension of q is found in three steps. The first step is to set the flag in C to record the value of $A(\langle e, x\rangle)$ in all $G_i(x)$. Let m be the value decided for $\Psi(C \oplus U \oplus Y, x)$. We set $A(\langle e, x\rangle)$ and all $G_i(x)$ equal to $1-m$. Next, a computation to make $\Delta(B \oplus Y, x) = 1-m$ is put into $B^{(e)}$. Let q_0 be the resulting extension of q.

Since $|x|$ is greater than $gap(s)$, clauses (3) and (4) in the description of E are satisfied. The remaining constraints imposed by I(Φ, $\overrightarrow{G}$, global) are explicitly respected. Those imposed by other type I strategies are respected using the default option of having no new strings in the relevant columns of A or B; those imposed any type II(Θ, Ω, i) strategy are respected (by clause (1.ii) of that strategy). The remaining properties needed to have an element of E are covered by having to have an element of U that is longer than $\ell(X)$ for each named set X and the extension of the coding for strategies of type IV and V. These we can satisfy as in Lemma 7.3, by extending $q(U)$ and defining the coding parameters to satisfy Formulas 6.12 and 6.14.

In the other case, I(Φ, $\overrightarrow{G}$, Ψ) reaches state (4) because Y is amorphous over p at some ℓ less than or equal to $in_{n+1}(s+1)$. Let ℓ and q be the least integer and extension of p in E such that q decides $Y \restriction \{0,1\}^{\leq \ell}$ but, for all G_i in $\overrightarrow{G}$, $p * q(G_i)$ does not decide $Y \restriction \{0,1\}^{\leq \ell}$. We may assume that $q(X)$ is equal to $p(X)$, for all X not equal to U and not in $\overrightarrow{G}$. In I(Φ, $\overrightarrow{G}$, Ψ), we build an extension of q as follows.

First, we extend $q(U)$ to include two new points x_1 and x_2 so that $2^{2^{2\ell_q(U)}}$ is less than $|x_1|$ respecting U's dominating the function associated with E, respecting U's dominating $u_\Phi \circ u_\Psi$ and satisfying $|x_2| > |x_1| > gap(s)$. By Lemma 7.1, this will be consistent with the resulting environment E. Then, we extend each $q(G_i)$ to be 0 on all strings of length less that $u_\Phi \circ u_\Psi(|x_1|)$. This is compatible with $E_0(s+1)$, because we still have $\ell(G_i) < |x_2|$.

Next, we extend $q(C)$ to be compatible with $p(C) * 0$. To obtain the inequality, we extend A and B to make $A(e)(x_1)$ equal to $\Delta(B \oplus Y, x_1)$ but unequal to the value forced

for $\Psi(C \oplus U \oplus Y, x_1)$. We set A and B equal to 0 on the remaining strings relevant to the conditional global constraints associated with type II strategies in $\overrightarrow{S}$ and of length less than or equal to $u_\Phi \circ u_\Psi(|x_1|)$. As above, we can then extend this condition to increase the domains of $r(X)$ for all X to respect the definite global constraints. Let r_1 be the resulting condition.

It remains to see that this action respects the conditional global constraints. By Corollary 7.2 and the choice of x_1 to have length greater than $gap(s)$, we are free to decide $A(\langle e, x\rangle)$ and then to extend the conditions on B and C, compatibly, and remain in E. For strategies of type II that do not change state or of type V, their only further constraint is included in U's dominating by f. We covered this point explicitly in the choice of x_1 and x_2.

The possibility of conflict occurs in the analysis of a type II conditional outcome. Suppose that II(Θ_0, Ω_0, i_0) is an element of $\overrightarrow{S}$ producing the state (1) environment. Since we did not set any flag in C, we have to check the conditions of (1.iii): for all z with $|z|$ less than or equal to $OUT(s)+out$, $\Theta_0(r(A)*0, z) = \Theta_0(p(s)(A)*0, z)$. Suppose that condition (1.iii) does not hold. Then, $r_1(A)$ must be incompatible with $p(s)(A)*0$; this can only happen when $r_1(A)$ has value 1 at $\langle e, x_1\rangle$ and compatible with $p(s)(A)*0$ elsewhere. This exactly meets the condition for II(Θ_0, Ω_0, i_0) to change state: to state (2), if I(Φ, $\overrightarrow{G}$, global) is of higher priority, and to state (3), otherwise.

Now consider the case of a finite outcome from in a type II strategy, II(Θ, Ω, i). In the case that II(Θ, Ω, i) enters state (2), it is because of an amorphous condition q relative to a higher priority strategy I(Φ_0, $\overrightarrow{G_0}$, Ψ_0). Let e_0 be the index of I(Φ_0, $\overrightarrow{G_0}$, global). There is a string x in $q(U)$ such that $|x| > gap(s)$, $\Phi_0(\overrightarrow{G_0})$ is amorphous over q at $\log(\log(|x|))/2$, and the value of $\Theta(A, z)$ depends on the value of A at $\langle e_0, x\rangle$.

In II(Θ, Ω, i), we construct a condition r as follows. Let Y_0 be the subset of $\{0,1\}^{\leq \ell}$ such that q forces $\Phi_0(\overrightarrow{G_0}) \upharpoonright \{0,1\}^{\leq \ell} = Y_0$. Let $\overline{Y_0}$ be $\{0,1\}^{\leq \ell} - Y_0$. Extend $q(B)$ to $r(B)$ by setting $r(B^{(e)})(\langle x_1, 1, Y_0, \overline{Y_0}\rangle)$ equal to 1 and making $r(B^{(e)})$ equal 0, for every other string of length small enough to be queried in the computation of $\Omega(B \oplus G_i, z)$. Define $r(G_i)$ to be compatible with $q(G_i)*0$ deciding all strings of length small enough to be queried in the computation of $\Omega(B \oplus G_i, z)$. Thus, we decide $\Omega(B \oplus G_i, z)$; let m be the value forced. We then let $r(A)$ be whichever of $p(s)(A)*0$ and $p(s)(A)*(A(\langle e_0, x\rangle)*0$ gives the inequality $\Theta_0(A, z) \neq m$.

Since $\Phi_0(\overrightarrow{G_0})$ is amorphous at ℓ over q, we find a condition $\overrightarrow{G_0}$ on the set of G_i named in p such that $\overrightarrow{G_0}(G_i) = r(G_i)$ and $\overrightarrow{G_0}$ forces that $\Phi_0(\overrightarrow{G_0})$ is incompatible with Y_0. For j not equal to i, we let $r(G_j)$ be defined on $\{0,1\}^{\leq \ell_r(G_j)}$ by

$$r(G_j) = \begin{cases} q(G_j) * 0, & \text{if } r(A)(z) = 1; \\ \overrightarrow{G_0}(G_j), & \text{otherwise.} \end{cases}$$

Thus, we force $A(\langle e_0, x\rangle)$ is equal to 1 if and only if Y is compatible with Y_0. By putting $\langle x_1, 1, Y_0, \overline{Y_0}\rangle$ into $B^{(e)}$, we force that Y is compatible with Y_0 if and only if $\Delta(B \oplus Y, x_1)$ is equal to 1. At other arguments, A and $\Delta(B \oplus Y)$ were given their default value 0.

This was the action taken by the type II strategy in state (2). First notice that the only possible element of $r(A) - p(s)(A)$ is $\langle e_0, x\rangle$ and $|x|$ is greater than $gap(s)$. Thus, the choice of r is compatible with clauses (3) and (5) in the description of E given in Lemma 7.1. For the type I strategies, we explicitly ensure that I(Φ_0, $\overrightarrow{G_0}$, global) is respected. The other type I strategies are respected by setting A and B to have their default values of 0. We can fill out r on the other parameters to satisfy the global constraints not of type II, as in Lemma 7.3.

If the condition r is not an element of the environment imposed by a higher priority type II strategy, we can repeat the argument given at the end of the analysis of the type I strategies to show a change of state in a type II strategy of higher priority.

If the type II strategy reaches state (3), the analysis is similar but much simpler. In particular, we do not have to worry about the type I strategy of index e_0 because it has lower priority.

The remaining finite outcome strategies, types III to V, are all similar. They require a diagonal step that is consistent with an appropriate redirection of the coding strategies. Further, they put no constraints on A, B or C and thereby avoid conflict with the strict controls put on by the type I and II strategies. We will give the analysis for a strategy of type IV as a canonical example.

For a IV(Θ, i, j)-strategy, a condition is constructed to strongly force

$$\Theta(G_i \oplus L \oplus U) \neq G_j. \tag{7.7}$$

We chose an x with $|x|$ greater than $f(\ell_{p(s)}(U))$ and greater than $gap(s)$, extend the conditions on all of the G_k with $k \leq_{P_N} i$, extend L to code the values in the extension

and to shunt the coding for all other $G_{i'}$ to the values of other $G_{j'}$, and set all other $\mathcal{G}$ conditions to be extended equally and satisfy Equation 6.12. Since $|x|$ is chosen so large and no values are specified for the remaining parameters, the only possible conflict is with a coding strategy of type IV. This is ruled out by the transitivity of P_N: there cannot be a sequence of coding constraints leading from G_j to G_i of type IV.

These are all of the cases for the finite outcomes. In each case, we gave a uniformly recursive method to find a condition so that either we would later discover a change of state or the condition would belong to the subsequently imposed environments. This is what is meant by the compatibility of $\vec{\mathcal{S}}$. ◊

Proof of Theorem 4.9: We can now complete the proof of Theorem 4.9. By Theorem 2.10, let $\mathcal{C}$ be a coherent recursive construction defining $gap(s)$ as in Definition 7.5, that eventually respects the initial strategy and all of the strategies of type I to V. Let A, B, C, D, L, P_0, Q_0, P_1, Q_1 and $\mathcal{G} = \{G_i \mid i \in N\}$ be the sets produced by $\mathcal{C}$. Since the strategies are uniform, $\mathcal{G}$ is a uniformly recursive set. By Theorem 3.6, let K_0 and K_1 be recursive sets such that

$$(K_0) \wedge (K_1) = (\mathcal{G}). \tag{7.8}$$

By the results of Section §6, all of the requirements of Section §5 are satisfied. This, together with Equation 7.8, is enough to conclude Theorem 4.9. ◊

§8. Further Results

Open Problems

Most of the global properties of the PTIME degrees are unknown. In this section, we list a few questions.

8.1. Question. (1) Is there a nontrivial automorphism of $\langle REC, \leq_p \rangle$ or of $\langle \mathcal{R}, \leq_p \rangle$?

(2) It would be very interesting to find a natural degree theoretic property that distinguishes one of the standard complexity classes in $\langle REC, \leq_p \rangle$. Is the ideal of the degrees of the DTIME($O(2^n)$) predicates definable without parameters in $\langle REC, \leq_p \rangle$? Is there a nontrivial ideal in that can be defined without parameters in $\langle REC, \leq_p \rangle$?

All of these questions would be answered by showing that the relation between $\vec{p}$ and x given by "$\vec{p}$ codes a standard model of arithmetic and a subset of the integers of that

model of PTIME degree x" is definable (in the appropriate model). We conjecture that this is the case.

We can give some weak evidence for a negative answer to (1) for $\langle \mathcal{R}, \leq_p \rangle$. Say that two predicates X and Y are *arithmetically equivalent*, if X can be defined in the structure N with an additional predicate for Y, and conversely. This is equivalent to saying that Y is recursive in a finite iteration of the Turing jump applied to X, and conversely.

8.2. THEOREM. *Suppose that ψ is an automorphism from $\langle \mathcal{R}, \leq_p \rangle$ to $\langle \mathcal{R}, \leq_p \rangle$. There is an a, such that for all x, if $a \leq_p x$ then the elements of $\psi(x)$ and those of x are arithmetically equivalent.*

PROOF: We adapt an argument from the context of the Turing degrees, see [8, Nerode-Shore]. As in Section §4, let M be a recursive set such that, for any predicate Z, if $M \leq_p Z$ then there is a sequence of parameters $\vec{P}$ coding P_N in $\langle \mathcal{R}, \leq_p \rangle$ and an exact pair H_0 and H_1 defining the set of integers in the coded model that represent elements of Z such that $\vec{P}$, H_0 and H_1 are PTIME in Z. Let m be the PTIME degree of M.

Suppose that $M \leq_p Z$ and z is the PTIME degree of Z. By the above, Z is coded in the structure of the PTIME degree below z. Since ψ is an automorphism, Z is also coded in the degrees below $\psi(z)$. The partial ordering of the degrees below $\psi(z)$ is arithmetically defined in terms of any element of $\psi(z)$. Thus, Z is arithmetically defined in terms of any element of $\psi(z)$. Applying the same argument to ψ's inverse ψ^{-1}, we see that if $\psi(z)$ is greater than m then any element of $\psi(z)$ is arithmetically defined in terms of any element of z. Thus, if z is greater than or equal to $m \vee \psi^{-1}(m)$ then the elements of z are arithmetically equivalent with those in $\psi(z)$. ◊

The interpretation of arithmetic given in Theorem 4.9 shows that the theory of $\langle REC, \leq_p \rangle$ is undecidable. However, the formulas used to show undecidability have a fairly complicated syntactic form. Perhaps a decision procedure can be given for a reasonable fragment of the theory.

8.3. QUESTION. Is the existential-universal theory of $\langle REC, \leq_p \rangle$ decidable?

We have not given any information about the metatheory of the PTIME many-one Turing degrees.

8.4. QUESTION. What is the Turing degree of the theory of the PTIME many-one degrees of the recursive sets?

REFERENCES

1. Ambos-Spies, K., *Minimal pairs for polynomial time reducibilities*, (to appear).
2. Cook, S. A., *The complexity of theorem proving procedures*, in "Proc. Third Annual ACM Sympos. on Theory of Compt.," 1971, pp. 151-158.
3. Groszek, M. J. and Slaman, T. A., *Foundations of the priority method I: finite and infinite injury*, (to appear).
4. Harrington, L. A. and Shelah, S., *The undecidability of the recursively enumerable degrees (research announcement)*, Bull. Amer. Math. Soc. (N. S.) **6** (1982), 79-80.
5. Harrington, L. A. and Slaman, T. A., *Interpreting arithmetic in the Turing degrees of the recursively enumerable sets*, (to appear).
6. Kleene, S. C. and Post, E. L., *The upper semi-lattice of degrees of recursive unsolvability*, Ann. of Math. **59** (1954), 379-407.
7. Ladner, R. E., *On the structure of polynomial time reducibility*, Jour. Assoc. Comp. Machinery **22** (1975), 155-171.
8. Nerode, A. and Shore, R. A., *Reducibility orderings: theories, definability and automorphisms*, Ann. Math. Logic **18** (1980), 61-89.
9. Shore, R. A., *The theory of the degrees below* $0'$, Jour. London Math. Soc. (Ser. II) **24** (1981), 1-14.
10. Slaman, T. A. and Woodin, W. H., *Definability in the Turing degrees*, Ill. Jour. of Math. **30** (1986), 320-334.
11. Soare, R. I., "Recursively Enumerable Sets and Degrees," Springer-Verlag, Berlin, 1987.
12. Spector, C., *On degrees of recursive unsolvability*, Ann. of Math. **64** (1956), 581-592.

Department of Mathematics; College of General Education; Nagoya University; Nagoya 464; Japan
Department of Mathematics; The University of Chicago; Chicago, IL 60637; U. S. A.